Pipeline Reconstruction from Fisheye Images

Yuhang Zhang
The Australian National University
yuhang.zhang@anu.edu.au

Richard Hartley
The Australian National University
richard.hartley@anu.edu.au

John Mashford
CSIRO
Australia
john.mashford@csiro.au

Lei Wang
The Australian National University
lei.wang@anu.edu.au

Stewart Burn
CSIRO
Australia
stewart.burn@csiro.au

ABSTRACT

Automatic inspection of pipelines has great potential to increase the efficiency and objectivity of pipeline condition assessment. 3-D pipeline reconstruction aims to reveal the deformation of the pipe surface caused by internal or external influences. We present a system which can reconstruct the inner surface of buried pipelines from multiple fisheye images captured inside the pipes. Whereas the pipelines are huge, a fatal defect can be as tiny as a fine crack. Therefore a reliable system demands both efficiency and accuracy. The repetitive patterns on the pipe surface and the poor illumination condition during photographing further increase the difficulty of the reconstruction. We combine several successful methods found in the literature as well as new methods proposed by ourselves. The proposed system can reconstruct pipe surface not only accurately but also quickly. Experiments have been carried out on real pipe images and show promising performance.

Keywords: 3D reconstruction, surface reconstruction, fisheye lens, water pipelines, pipe inspection, image processing.

1 INTRODUCTION

Water pipelines are indispensable facilities of modern urban systems. After serving for decades underground, the condition of the pipelines deteriorates to varying degrees. Timely inspection and repair is therefore required to prevent imminent collapse. Traditionally pipe inspection involves intensive manual effort. Manual image interpretation is an expensive process for which wrong decisions caused by fatigue and subjective bias are inevitable. Hence a computer-aided inspection system is of great value.

We present a system which can reconstruct the inner surface of buried water pipes based on a sequence of images captured inside the pipes (Figure 1). Deformation of the pipe surface which foreshadows the pipeline collapse can then be detected from the reconstructed model. Early work on similar applications relied on range cameras such as laser scanners, which is expensive. Later, due to the developments of computer vision, methods solely based on 2D images were proposed [3, 8, 9]. However, because of the limitation in computer vision at the time and the difficulty in this particular application, some of these works made restrictive assumptions such as that, the pipes are built from bricks which provide distinctive patterns; and the others terminate with reconstructing a small group of isolated points only. Some 3D reconstruction applications of general scenes [18] bear the same limitation as well. More recently, several large-scale 3D reconstruction applications of general scenes have been proposed [1, 20], which can reconstruct millions of points in a relatively short time. However, their implementation requires high-end parallel computers.

What distinguish the proposed system from all the previous ones are:

1. we intensively reconstruct the pipe surface, which is composed of millions of points, rather than a group of selected points;
2. our algorithm is fast and can be implemented on normal PCs;
3. we have proposed a number of specific mechanisms to increase the robustness of the system, so that it can work with pipe surface without distinctive patterns under poor illumination conditions.

We first give an overview of the reconstruction problem, as well as our method. When discussing each step in detail, experimental results will be provided accordingly. As will be seen, our method performs not only accurately but also quickly.

2 OVERVIEW

We make no particular assumption about the material of the pipelines. Actually, the pipes a civil engineer frequently confronts are made from concrete which gives

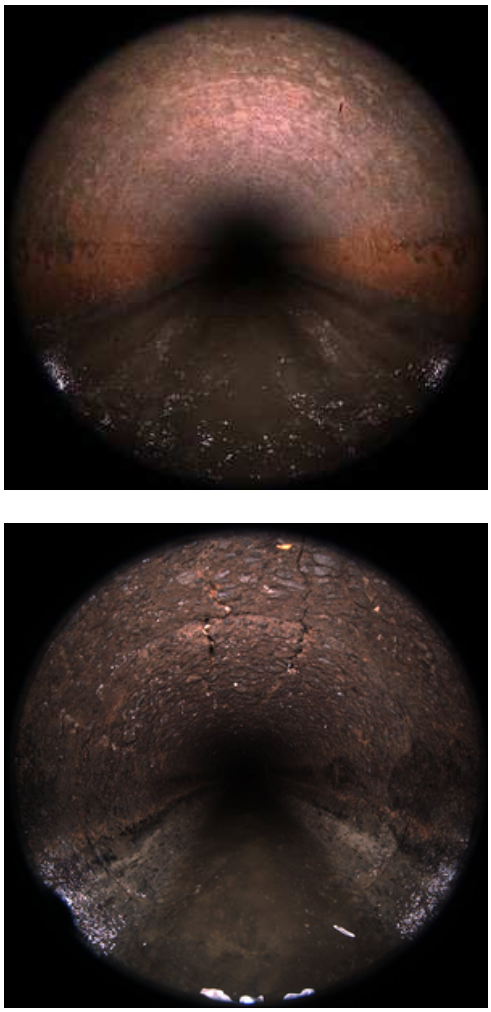

Figure 1: fisheye images captured inside of the pipelines. The pipe on the top is in relatively good condition, whereas the one on the bottom is in poor condition.

little reliable texture. We will therefore make our system capable of handling pipes of this type.

To assess the pipeline condition, images are collected by a mobile camera travelling through the pipelines. To capture more details from the pipe surface, the mobile camera is equipped with a wide view angle fisheye lens rather than an ordinary perspective lens. During photographing the illumination is provided by a light fixed to the camera, which can only illuminate a limited range in the pipe unevenly. Figure 1 shows two example images captured in different pipelines. As we can see, only the peripheral regions of the images contain clear pipe surface. The texture on the pipe surface is fine and weak. In the same pipe the surface appears to be similar everywhere. A sequence of images is captured as the camera moves. Two adjacent images share overlapping regions.

Our reconstruction follows a four-step paradigm.

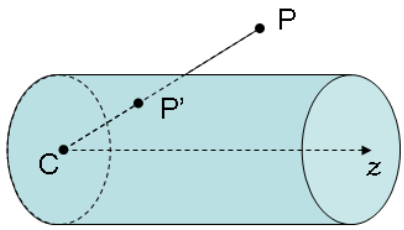

Figure 2: image cylinder: **C** is the camera center; *z* axis is the central axis of this image cylinder; **P** is an object point, with **P'** as its image.

1. Firstly, initial point matching is established between overlapping images;
2. Matched points will then be utilized to estimate the relative pose of calibrated cameras corresponding to different views.
3. With the obtained camera parameters, we implement dense matching between overlapping images while enforcing the epipolar constraints. This step was never included in the previous works [3, 8, 9], in which only those matched points detected in the first step were reconstructed. This step is also arguably the most sophisticated and time consuming step in the whole algorithm. Handling it efficiently is our major contribution.
4. Finally, the 3D location of each point in the image is densely determined through triangulation and a 3D model is built up.

As computer vision algorithms about perspective cameras have already been well studied [4], one might transform each fisheye image to a perspective image to simplify the subsequent process [8, 9]. However, such a transformation either produces an extremely large perspective view which significantly upsampled the peripheral region in the original image, or produces a perspective view of proper size but at the cost of cropping off the peripheral region. In either case, we will destroy the region which really contains the important information in the original image. Therefore, in our work we choose to process the images in their original form, or transform them, when necessary, onto an image cylinder (Figure 2) instead of an image plane.

We define an image cylinder by specifying its central axis. Its radius can be deliberately set to unity without affecting its functionality. The central axis of the image cylinder can be the optical axis of the camera or the baseline between two cameras, depending on the circumstances. The cylindrical image of each point in the 3D world is generated by the intersection of the image cylinder and the ray going from the point to the camera center. Each parallel line on the cylindrical surface

functions like a perspective camera by itself, however, altogether they receive an omnidirectional image more readily than a normal perspective camera does. This image cylinder is particularly useful during point matching and depth estimation, as will become clear soon.

In the remaining part of this paper, we discuss each step mentioned above in detail.

3 INITIAL POINT MATCHING

Due to the development of local invariant features [13, 16], finding corresponding points between overlapping images is much easier now than ever. Comprehensive surveys into the feature detectors and descriptors can be found in [14, 15]. However, point matching on a pipe surface is still difficult due to the faint and similar patterns everywhere. Moreover, whereas all the proposed local invariant features are approximately invariant under affine transformations, the transformation conducted by a fisheye lens is not even perspective, but nonlinear. Thus the corresponding points identified by local invariant features on pipe surface contain many false matches. Our experiments show that the number of false matches can easily exceed the number of true matches by an order of magnitude.

To improve the situation, besides enforcing loose geometry constraints, we transform each fisheye image onto the image cylinder we discussed in Section 2. The image cylinder here takes the optical axis of the camera as its central axis. The consequential advantage of such a transformation is obvious. Since the optical axis of all cameras are roughly parallel to each other as well as to the central axis of the pipe, the images generated on different image cylinders only differ from each other approximately by a simple translation. Comparing to the original fisheye images, we not only remove the scale difference between corresponding regions in different images, but also largely rectify the distortion caused by the nonlinear projection through a fisheye lens. Hence the corresponding points found by local invariant features on the cylindrical images are more reliable. Geometry consistency is also easier to enforce on the transformed images. All line segments connecting corresponding points in two cylindrical images should be roughly parallel and of almost the same length. After detecting corresponding points in the transformed images, we can easily back-project them onto the original fisheye images to facilitate camera pose estimation.

Figure 3 shows the matching results on the original images and the transformed images respectively. Particularly, Hessian-affine detector [16] and SIFT descriptor [13] are used for feature extraction. Matches are identified if two SIFT features share a Euclidean distance under a predefined threshold. Although point matching is between two images, we only present one of them here for clear presentation. We plot the matched points from two images onto one image and connect each pair with a yellow line segment. As we can expect, the lines in the original image should all roughly point to the image center, whereas those in the transformed images should all be roughly horizontal. We only present those matches that can be verified with these loose geometry constraints in Figure 3. On the original image 239 matches passed the verification, whereas on the transformed image 563 matches passed the verification. That justifies our earlier discussion that matching on the transformed images is more reliable.

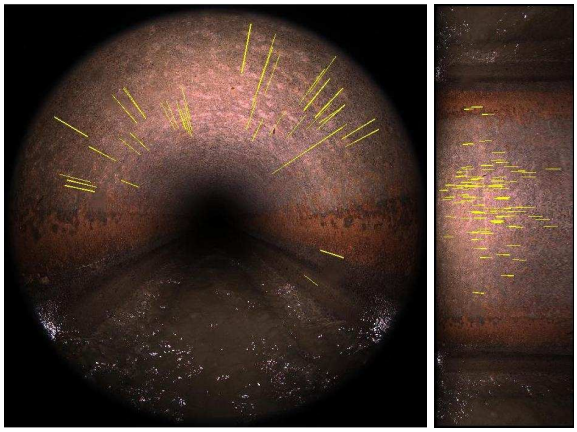

Figure 3: matches found on the image pair of original form and transformed form respectively. Only those matches that pass the loose geometry verification are presented. The rejected false matches are thousands in number. They happen so frequently because the pipe surface is similar everywhere.

Intuitively the matches from both cases are more than sufficient to implement subsequent estimation. One might therefore suspect the necessity of the cylindrical transform. However, as we can see, the lines presented in the image do not seem to match the numbers given above. That is because more than one match can happen intensively on neighbor pixels. Considering matches at the same location does not increase the estimation accuracy, more matches than sufficient is in fact necessary. The number of qualified matches also depends on the texture on the pipe surface. On some smooth surfaces, the number of matches will be much smaller as fewer local features can be detected. That is when the image transformation becomes more important.

Some false matches still remain in Figure 3 as their line segments are not of reasonable length. Again, it is more convenient to enforce this constraint on the transformed images rather than on the original images. On the transformed images, the length of all lines segments should be roughly equal. In the original images, their length should not be equal due to the nonlinearity of the fisheye lens, which is difficult to use as a loose constraint.

4 CAMERA POSE ESTIMATION

A calibration method for fisheye cameras can be found in [10]. Here we assume the camera is calibrated and only aim to estimate the external parameters of the camera. We have briefly discussed the reason why we do not transform the original image into perspective view in Section 2. Particularly on camera pose estimation, the nonlinear transformation between a fisheye view and a perspective view might significantly enlarge the matching error from one pixel to hundreds of pixels in the peripheral region of the image. Hence we need a pose estimation scheme that can be applied directly to fisheye images and is efficient.

We use a modified version of the direct estimation method initially designed for perspective cameras [6]. The main result of the original method is that, given a close enough initialization of the camera parameters as well as the point locations, a structure from motion problem can be solved directly using some iterative optimizing algorithms, e.g. the Levenberg-Marquardt algorithm [12]. The advantage of this method lies in the fact that it is one-stop. It requires no sophisticated operation on any interim variables, e.g. the fundamental matrix required in [5] or the measurement matrix required in [21]. The disadvantage of this algorithm is the requirement of a close initialization, which is usually impossible, especially when the number of unknown parameters is huge.

We discover that the advantage and the disadvantage of the direct estimation method can be magnified and reduced respectively in our problem. In particular, unlike the other algorithms of structure from motion applications, this algorithm bears no assumption on the camera model, neither perspective nor affine. That means it can be adapted to fisheye camera as well, as long as we change the cost function in the Levenberg-Marquardt minimization from the perspective projection to the fisheye projection. Furthermore, as we know the normal condition of the pipelines as well as the approximate location of the camera with respect to the pipe, we can initialize all the parameters accordingly. Obviously, many other inspection purposed applications share the same convenience.

Another important fact about the parameter initialization is that, the parameters are not independent. More precisely, from the parameters of two random cameras, we can accurately determine the 3D locations of all the matched points captured by the two cameras through triangulation. This observation largely reduces the number of variables we need to initialize, i.e. we only initialize the camera parameters, and then derive the location of the points. Besides dependence, obviously, there is also independence between different parameters. Whereas millions of points were captured from thousands of different locations, the camera pose for each image is only related to the dozens of points that have appeared in its image. The location of each point is only affected by the few cameras capturing it. This observation not only leads to the simplification within Levenberg-Marquardt optimization, i.e. the sparse-Levenberg-Marquardt [6], but also to the simplification of our reconstruction. We firstly estimate the camera parameters and points location locally between each pair of adjacent images with the direct estimation method. Although this estimation is local, it has already considered most of the information relevant to the two cameras. Hence the output should still be quite accurate. We then transform all the estimated points and cameras into the same frame of reference. That gives us the initialization of a global direct estimation. Indeed, when the global consistency is not compulsory, we can even terminate without a global estimation. Later we will see, at least for the purpose of pipe condition assessment, local estimation can already detect deformation and cracks on the pipe surface.

The error to be minimized with Levenberg-Marquardt algorithm is given by (1), where $\hat{x}_{ij}$ is the coordinates of point i observed in image j, and x_{ij} is the estimated coordinates of the corresponding point in the corresponding image. When $\hat{x}_{ij}$ is unknown, which really means point i is not observed in image j, we set $\hat{x}_{ij} = x_{ij}$, so that their difference is 0 and the total error will not be affected. During local estimation, as the numbers of points and cameras are limited, the sparse-Levenberg-Marquardt algorithm converges quickly. In our experiment, it takes about 0.5 seconds to estimate the relative pose between each pair of cameras, when 200 point matches are involved. The root mean square error is around one pixel upon converging.

$$e = \sum_j \sum_i ||\hat{x}_{ij} - x_{ij}||^2 \tag{1}$$

Further more, we might add the intrinsic camera parameters into the local estimation. That converts our problem to an uncalibrated reconstruction, requiring inputting three images each time. We do not recommend estimation based on three views. That is because the number of matching points that can survive three views are usually too small to facilitate reliable estimation.

5 INTENSIVE MATCHING

Whereas reconstructing a set of isolated points is sufficient to reveal the pipe deformation on large scale, intensive points reconstruction is required to reveal those cracks which are only several pixels wide on images. To intensively reconstruct the pipe surface, we need intensively match the points on the pipe surface.

Implementing intensive stereo matching between overlapping images is by nature a difficult problem, even though we can narrow the matching range using

the epipolar constraint. A good review of relevant algorithms can be found in [19]. The state of the art of intensive stereo matching lies in the α-expansion method proposed in [22], which approaches the problem by way of optimizing a multi-label Markov Random Field (MRF). However, when the size of the image is huge, optimizing a corresponding MRF requires heavy computation. Another method called FastPD [11] is faster but requires much more memory. More recently, a hierarchical mechanism is incorporated into MRF optimization [23], enabling optimizing large MRFs more efficiently with low memory occupancy.

However, due to the following reasons, our problem cannot be solved by these off-the-shelf methods. Firstly, since the light source is carried by the moving camera, corresponding points in different images are captured under significantly different illuminations, which obviously makes the matching tougher. Secondly, even the hierarchical mechanism [23] largely boosts the speed of solving an individual problem, intensively matching a large number of images is still a huge task. Therefore, we propose two mechanisms to improve the situation.

5.1 Illumination Regularization

Some illumination invariant description and comparison methods have been proposed in the literature, such as the Normalized Cross-correlation (NCC) and the SIFT descriptor [13]. They non-exclusively require more complex computation, which will significantly slow down the system. Here instead of using illumination invariant description, we make the illumination invariant.

Although the light source moves during photographing, its relative position to the camera center is fixed and the location of the camera center within each cross-section of the pipe is in general stable. That suggests, the pipe surface captured by the pixels on the same location within every image is illuminated by approximately the same light. From each pipeline, we have collected thousands of images. The average grey level of a pixel on the same location over thousands of images can be then regarded as the illumination intensity of this pixel or its corresponding points on the pipe surface.

Figure 4 shows the average illumination intensity on images captured in the two pipelines. They are different because the camera travelled at different height in the two pipes and the deterioration degree of the two pipes are different. Based on the illumination intensity images in Figure 4, we can regularize the illumination within each image through (2), where $I(i)$ is the pixel value of pixel i in the original image, $G(i)$ is the grey level of pixel i in the illumination intensity image, a is a positive constant controlling the brightness in the regularized image. Figure 5 compares the image before and after illumination regularization. Especially on the regularized cylindrical images, the obvious illumination variance is removed leaving all pixels under comparable illumination. After illumination regularization, we can easily measure the similarity between pixels by the absolute difference between their regularized pixel values.

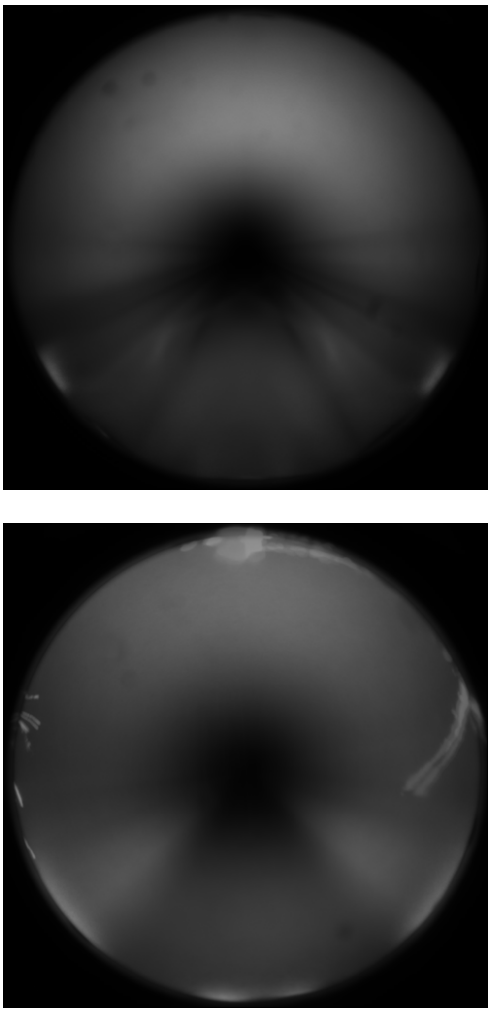

Figure 4: the average illumination intensity obtained from images of two pipelines: some dark blobs can be observed on the top image, which were caused by water drops spread onto the lens; the white threads on the bottom image are caused by some rubbish attached to the lens. However, their affect to the matching process is ignorable.

$$I_r(i) = \frac{aI(i)}{G(i)} \tag{2}$$

5.2 Sequential MRF Optimization

We first explain the design of α-expansion as well as its hierarchical version in our problem and then intro-

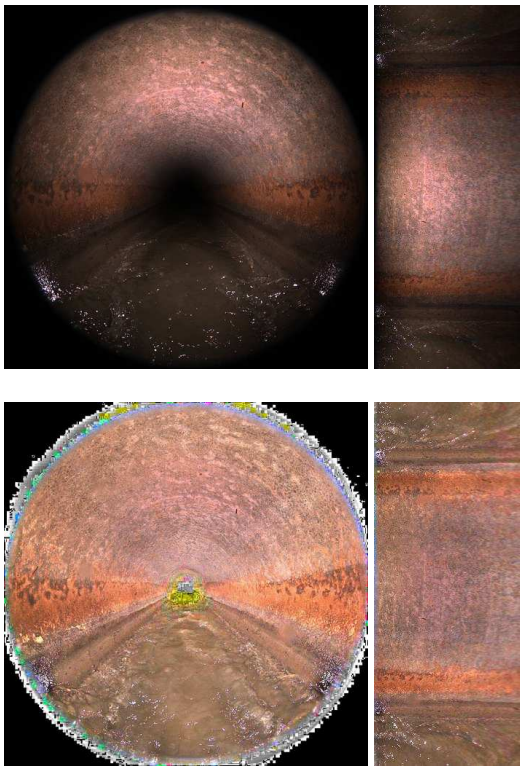

Figure 5: images before and after illumination regularization.

duce our sequential mechanism, which further boosts the speed of our system.

α-expansion After rectification [4], the corresponding points between two overlapping images all lie in corresponding scan lines. One of the two images will later be referred to as the reference. Localizing corresponding points along a scan line, namely estimating the disparity of each object point within the two images can be modelled as estimating the variables in a second order Markov Random Field.

In particular, each variable in the MRF corresponds to a pixel in the reference image. The value of each variable corresponds to the disparity of its corresponding pixel. The probability for each variable to have a particular value, or equivalently for a pixel to have a particular disparity, is subject to two factors. The first one, a function of the color difference between the two pixels related by this particular disparity, is usually referred to as the unary term or the data term. In our work, we use the following unary term:

$$U_i = ||I_1(i) - I_2(i')||_1 \tag{3}$$

where $I_1(i)$ is the color of pixel i in the reference image, $I_2(i')$ is the color of the pixel related to i by its current disparity in the other image, and U_i is computed as a L_1-norm difference between the two. The other one, a function of the disparity difference between the pixel and its neighbor, is usually referred to as the binary term or the smooth term. Each pixel usually has four neighbors, hence there are four binary terms. Binary terms are used to enforce the smooth constraint, i.e. the disparity of points in a scene should be smooth almost everywhere [17]. In our work, we use the following binary term:

$$B_{ij} = |L(i) - L(j)| \tag{4}$$

where $L(i)$ and $L(j)$ is the disparity of two neighbor pixel i and j in the reference image, and B_{ij} is computed as their absolute difference.

The unary term and the binary term really play the role of likelihood and prior in the Bayesian theory. Therefore, through maximizing the probability of a MRF, one really globally maximizes the posterior of each variable and obtains the most probable disparity of each pixel. Due to the Hammersley-Clifford theorem, maximizing the joint probability of variables in the above MRF is equivalent to minimizing the following cost-function:

$$E = \sum_i U_i + \lambda \sum_{ij} B_{ij} \tag{5}$$

where λ is a positive constant balancing the weight between the unary term and the binary term. An effective way of perceiving (5) is through constructing a weighted graph. As shown by Figure 6, each vertex in the graph corresponds to a pixel in the reference image or a disparity value. Edges are created between each disparity vertex and all the pixel vertices. Each edge of this type can be represented by a term U_i in (3). Pixel vertices which are neighbors in the image are connected by edges as well. Each edge of this type corresponds to a B_{ij} in (4). Then, minimizing (5) is equivalent to finding the minimal cut on its graph after which each subgraph contains one and only one disparity vertex.

If the graph contains only 2 disparity vertices, the minimal cut can be found using the max-flow algorithm, regarding the two disparity vertices as the source and the sink respectively. When the number of disparity vertices is larger than two, minimizing (5) is in general NP-hard [2]. α-expansion can provide a high quality suboptimal solution in polynomial time.

Starting from a random initial state, α-expansion sequentially examines the applicability of each disparity, represented by α, to all the pixels. In particular, for each α, a new graph is created. In the new graph, the source node corresponds to the current disparity of each pixel; the sink node corresponds to the α disparity. Those pixels, whose current disparity is α are not included into the new graph. A bi-cut is then implemented using max-flow algorithm to determine whether the pixels currently having other disparities should change their disparities to α. After each

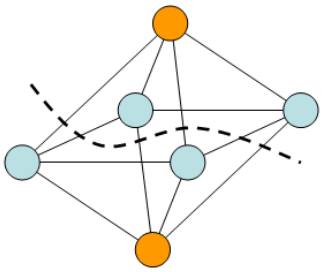

Figure 6: a graphic explanation of minimizing (5): the four blue vertices each correspond to a pixel in the reference image; the two orange vertices correspond to two possible disparities respectively; minimizing (5) is equivalent to a minimal cut to the graph after which each subgraph contains one and only one disparity vertex. The dashed line in the figure shows a possible cut.

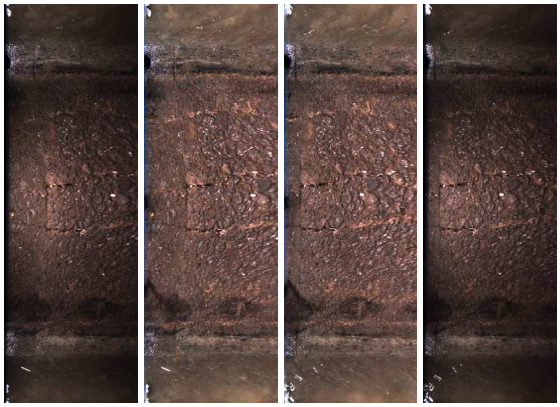

Figure 7: two adjacent images mapped onto the same image cylinder. Images before and after illumination regularization are both provided for comparison.

round of bi-cut, only the subgraph containing α vertex can be increased. That is why the algorithm is named as α-expansion. To compensate the loss in optimality, multiple outer iterations are usually implemented.

Denote the number of outer iterations as m, the number of disparity vertices as n, the processing time of max-flow algorithm as f. The processing time of α-expansion is mnf. When running on images of small size, *e.g.* 300×300, α-expansion can usually terminate quickly in 20 seconds on a normal PC. However, when dealing with a pair of images in large size, whose disparity range is usually large as well, the max-flow algorithm needs to be implemented on a huge graph many times. The processing time of α-expansion expands significantly. Our experiments show that when dealing with a stereo pair in the size of 1000×1000, α-expansion needs more than 30 minutes to converge. That is by definition too slow for practical use. The alternative method, FastPD, cannot be applied either, because a normal PC cannot provide sufficient memory space.

Hierarchical α-expansion The idea of hierarchical α-expansion can be explained as solving the problem with α-expansion under a low resolution first, and then fine tuning the low resolution solution onto higher resolution through optimizing another MRF. More details can be found in [23]. As these two steps can be implemented recursively, the original problem is really solved in a coarse-to-fine manner. Besides, since the MRFs being optimized in the two steps are both much smaller than the original one, the processing speed is largely improved. With the hierarchical α-expansion, processing a stereo pair in the size of 1000×1000 requires only around 10 seconds on a normal PC, and the optimality is comparable to the original α-expansion.

Figure 7 shows two sample images on which we have implemented hierarchical α-expansion. This time, the central axis of the image cylinder is the baseline connecting the two camera centers. As we have already obtained the external parameters of the cameras, we can accurately generate the cylindrical image through back-projection. Although this image cylinder is different from an image plane in shape, it can parallelize the epipolar lines as well. It takes minor effort to snip and unwind that cylindrical image into a planar image. Just make sure to snip the two cylinders along the same epipolar line. So we obtain an image pair in the form people usually deal with during intensive matching, namely corresponding points always lie on the same scan line. The pipe surface presented in these two images contains a vertical connection line and two horizontal narrow cracks, which will test our algorithm's capability in detecting small defects on the pipe surface. We crop off the region submerged by water before implementing graph cuts. That is because we are only interested in the pipe surface, and that dropping the water region can help saving processing time. Figure 8 shows the interim and final results of the hierarchical graph cuts. We can see how the final disparity map is reached through a coarse-to-fine procedure. The disparity value is larger in the center of the image, which corresponds to the top region in the pipe. That suggests that the camera is closer to the top of the pipe compared to the left and right sides of the pipe. The vertical connection line and the horizontal cracks can be clearly observed in the final result as well.

Sequential α-expansion To further boost the processing speed, we propose a sequential mechanism in MRF optimization, the key idea of which lies in better label initializations and smaller label range. The time cost by the max-flow algorithm which is a subroutine in α-expansion depends on the flows needed to be pushed before reaching the optimal state. The number of necessary flows depends on the initial state of the network. That really suggests, if the initial state of the network is

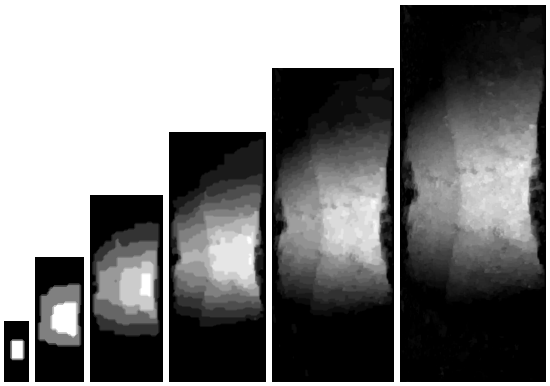

Figure 8: the interim and final results of the hierarchical graph cuts.

more similar to the optimal state, fewer flow, and hence less time, will be needed in optimization. Moreover, starting from an initial state close to the optimal also reduces the number of outer iterations in the α-expansion algorithm. A smaller label range will reduce the number of max-flow implementations in a single iteration.

Whereas for a contextless image pair one can only initialize all labels to be zero or arbitrary values, for sequential pipe images in our case we can largely predict the label configuration. The disparity of each point on the pipe surface is determined by two factors: firstly, its deterioration degree; secondly, and more importantly the location of the camera center. If the camera center travels along the central axis of the pipe, the disparity of different points will only differ slightly due to deterioration. However, if the camera center travels along some line far away from the central axis, the disparity of different points on the pipe will vary significantly. Although the deterioration degree of different regions on the pipe surface is arbitrary, the location of the camera center within each cross section of the pipe is generally stable. Therefore, we only use a large label set during the intensive matching for the first few image pairs. We can then acquire the relative location of the camera within the cross-section of the pipe, or more directly the average disparity along each scan line in the image. On subsequent image pairs, the pixels on each scan line are initialized with the corresponding average disparity. A smaller label set will then be used to estimate their disparities accurately. The smaller label set only needs to cover the variety caused by deterioration, which will be significantly narrower than that caused by the camera location. The MRF optimizing speed is hence boosted.

6 BUILDING A 3D MODEL

Through dense matching on the image cylinder, we have acquired the depth information related to each pixel on the cylindrical image. Together with the camera parameters estimated earlier, we can easily determine the 3D location of each point on the pipe surface through triangulation. The scale ambiguity is removed by setting the length of the baseline between two camera centers as unity. From each pair of adjacent images, we can obtain several millions of isolated 3D points. For better visualization, we might reconstruct a continuous surface with these isolated points using the algorithm proposed in [7]. However, a model containing millions of independent points is too huge for a normal PC to render.

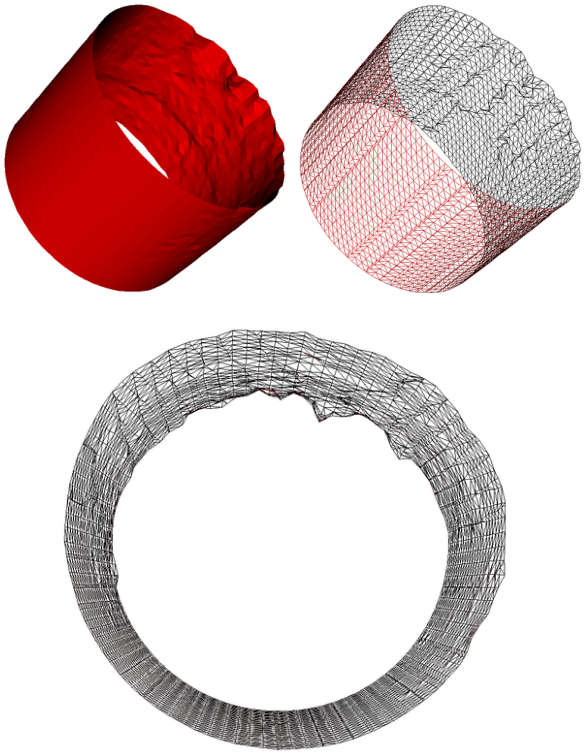

Figure 9: reconstructed pipe surface from the point cloud together with its triangulation state.

Figure 9 only shows the surface reconstructed from one hundredth of all the points. However, even after this significant downsampling, the connection line is still clearly presented, so is the pipe deformation on the large scale. The two cracks are missing because they are both less then ten pixels wide, which can not be preserved during this one hundredth downsampling. However, their existence and state have been represented by the point cloud containing millions of independent points, which will be assessed by civil engineers during force analysis. Note that we can only reconstruct the pipe surface above the water. We observe a complete cylinder here because the missing part has been manually complemented with ideal cylindrical surface.

7 CONCLUSION

We successfully reconstruct the inner surface of buried pipelines from a sequence of fisheye images. The obtained point cloud can be used to generate a virtual surface for visualization, as well as to facilitate other algorithms for pipe condition analysis. We used various efficient and reliable schemes over the four-step reconstruction. We paid particular attention to the process of intensive matching, which is generally slow and memory demanding based on previous algorithms. Our new method overcomes the obstacle of illumination variance and largely boosts the speed. More improvement on 3D model generation is still necessary. One possible development lies in automatically detecting regions of interest and unevenly downsampling the point cloud accordingly. This will be a direction of future work.

ACKNOWLEDGEMENTS

This work was supported by CSIRO, Water for a Healthy Country Flagship.

REFERENCES

[1] Sameer Agarwal, Noah Snavely, Ian Simon, Steven M. Seitz, and Richard Szeliski. Building rome in a day. In *ICCV*, pages 72 –79, 2009.

[2] Endre Boros and Peter L. Hammer. Pseudo-boolean optimization. *Discrete Appl. Math.*, 123:155–225, November 2002.

[3] D. Cooper, T.P. Pridmore, and N. Taylor. Towards the recovery of extrinsic camera parameters from video records of sewer surveys. In *MVA*, pages 53–63, 1998.

[4] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, March 2004.

[5] Richard I. Hartley. Estimation of relative camera positions for uncalibrated cameras. In *ECCV '92: Proceedings of the Second European Conference on Computer Vision*, pages 579–587, London, UK, 1992. Springer-Verlag.

[6] Richard I. Hartley. Euclidean reconstruction from uncalibrated views. In *Applications of Invariance in Computer Vision*, pages 237–256, 1993.

[7] Hugues Hoppe. *PhD Thesis: Surface Reconstruction from Unorgnized Points*. 1994.

[8] J.H. Kannala, S.S. Brandt, and J. Heikkila. Measuring and modelling sewer pipes from video. In *MVA*, volume 19, pages 73–83, March 2008.

[9] Juho Kannala and Sami S. Brandt. Measuring the shape of sewer pipes from video. In *MVA*, pages 237–240, 2005.

[10] Juho Kannala and Sami S. Brandt. A generic camera model and calibration method for conventional, wide-angle, and fisheye lenses. *PAMI*, 28(8):1335–1340, 2006.

[11] N. Komodakis and G. Tziritas. Approximate labeling via graph cuts based on linear programming. *PAMI*, 29(8):1436–1453, August 2007.

[12] K. Levenberg. A method for the solution of certain non-linear problems in least squares. *Quarterly Journal of Applied Mathmatics*, II(2):164–168, 1944.

[13] David G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004.

[14] K Mikolajczyk and C Schmid. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(10):1615–1630, October 2005.

[15] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Van Gool. A comparison of affine region detectors. *Int. J. Comput. Vision*, 65:43–72, November 2005.

[16] Krystian Mikolajczyk and Cordelia Schmid. Scale & affine invariant interest point detectors. *IJCV*, 60(1):63–86, 2004.

[17] Tomaso Poggio, Vincent Torre, and Christof Koch. Computational vision and regularization theory. *Nature*, 317:314–319, September 1985.

[18] Radka Pospíšilová. Occlusion detection and surface completion in 3d reconstruction of man-made environments. In *WSCG*, 2007.

[19] Daniel Scharstein and Richard Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *Int. J. Comput. Vision*, 47:7–42, April 2002.

[20] N. Snavely, S.M. Seitz, and R. Szeliski. Skeletal graphs for efficient structure from motion. In *CVPR*, pages 1 –8, 2008.

[21] Carlo Tomasi and Takeo Kanade. Shape and motion from image streams under orthography: a factorization method. *Int. J. Comput. Vision*, 9(2):137–154, 1992.

[22] R. Zabih, O. Veksler, and Y.Y. Boykov. Fast approximate energy minimization via graph cuts. In *ICCV*, pages 377–384, 1999.

[23] Yuhang Zhang, Richard Hartley, and Lei Wang. Fast multi-labelling for stereo matching. In *ECCV 2010*, volume 6313, pages 524–537. Springer, 2010.