A study in the topology-aware reconstruction of thin tubular structures

Juan Montes

A thesis
in
The Department
of
Computer Science

Presented in Partial Fulfillment of the Requirements
For the Degree of Master of Computer Science
Concordia University
Montréal, Québec, Canada

January 2015

CONCORDIA UNIVERSITY
School of Graduate Studies

This is to certify that the thesis prepared

By: Juan Montes

Entitled: A study in the topology-aware reconstruction of thin tubular s

and submitted in partial fulfillment of the requirements for the degree of

Master of Computer Science

complies with the regulations of the University and meets the accepted standards with respect to originality and quality.

Signed by the final examining committee:

Dr. Nikolaos Tsantalis Chair

Dr. Thomas Fevens Examiner

Dr. Peter Grogono Examiner

Dr. Tiberiu Popa Supervisor

Approved by ______________________________
Chair of Department or Graduate Program Director

____
Dean of Faculty

Date ______________________________

Abstract

A study in the topology-aware reconstruction of thin tubular structures

Juan Montes

This thesis is dedicated to the 3D reconstruction of thin tubular structures, such as cables or ropes, from a given image sequence. This is known to be a challenging task, mainly because of self-occlusions of the structure and its fine details. This new approach combines image processing tools with physics simulation to faithfully reconstruct jumbled and tangled cables in 3D. This method estimates the topology of the tubular object in the form of a single 1D path and also computes a topology-aware reconstruction of its geometry. This method is evaluated on both, synthetic and real datasets and demonstrate that this method favourably compares to state-of-the-art methods.

Acknowledgments

I would like to thank my family for their unconditional support which today results in the finalization of this thesis. To my mother, for lifting me up when I needed it. To my father, who transmitted me his passion for this field. To my sister, for her emotional support. Finally, I would like to thank my supervisor for showing me this road and helping me develop my work.

Contents

List of Figures

List of Tables

Dedicated to my father.

Chapter 1

Introduction

3D reconstruction deals with the problem of finding a representation of a shape given a set of images or depth information. Among these shapes, thin structures pose a special challenge since the volume obtained through traditional reconstruction methods is not enough to get an accurate representation of the shape. Thus, the reconstruction of such structures is an open problem.

Tubular structures occur in a variety of instances, such as electric cables, fire and garden hoses, and ropes, among many others (fig 1.1). While any of these examples share the property of being a deformed tube, the way they are arranged to themselves, i.e., the way they bend, overlap, twist, or self-occlude, is different. Therefore, it is difficult to correctly reconstruct the geometry of these varying topological arrangements[1].

While the field of 3D surface reconstruction has made impressive progress over the last few years, conventional reconstruction methods are challenged in this context as tubular objects can be relatively thin. Video based reconstruction methods such as structure from motion (Wu et al. [1], Wu [2]) provide limited quality even when many images are being accumulated as shown in Figure 1.2. Emerging colour and depth cameras such as the Kinect device have paved the way for a more detailed reconstruction compared to conventional colour cameras (Zhou and Koltun [3], Izadi et al. [4]). However, while significant progress has been made, the depth quality of current sensors is not sufficient to reconstruct thin features. An important limitation common to all methods mentioned is that they do not take the topology of the object into consideration. However, understanding the topology is an important prior in the reconstruction process as illustrated in Figure 1.2.

[1] In the context of computer vision, topological arrangements and topological consistency means that the reconstruction preserves the structure of the object being reconstructed.

Figure 1.1: Image of a bunch of cables. It can be observed how they self-occlude each other and how they overlap in a simple configuration.

In the case of 3d reconstruction of thin nearly one dimensional structures, several methods have been proposed in the form of 1D poly-lines such as arterial snakes (Li et al. [5]) and 3D skeletonizations such as curve skeleton extraction (Tagliasacchi et al. [6]) and L1-medial skeleton (Huang et al. [7]).

The method proposed reconstructs a tubular structure from a set of images. First, the 3D camera positions of each image are estimated using Structure from Motion (Wu et al. [1], Wu [2]). The images are segmented by colour and the junction areas are estimated in 2D. Then, the camera information is used along the segmented images with the estimated junctions to compute a fused volumetric grid in 3D. From this grid, the parts of the cable where it does not overlap are estimated. Afterwards, these parts are smoothed and corrected using a physics simulation approach. The paths are then connected where the cable overlaps. Finally, The crossings of the cable are smoothed using the physics simulator once again.

This thesis introduces a new reconstruction pipeline with two new contributions to current state of the art methods that combines image processing techniques with physics simulation for high fidelity reconstruction of thin tubular objects. The first contribution robustly fuses the data from multiple images to identify the different segments of the cable as well as their 3D topological structure. While this can be achieved using fairly standard image processing techniques for individual images, robust fusion of the 2D

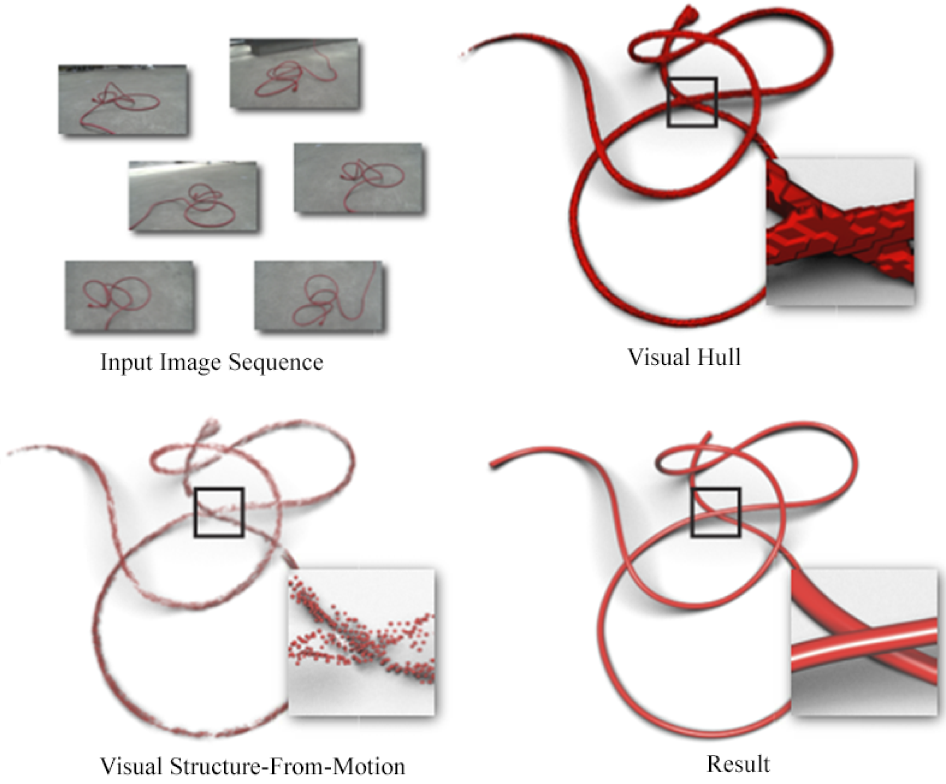


Figure 1.2: Given a sequence of colour images of a tubular structure as input, this method computes the geometry of the tubular structure. While visual hull and visual structure-from-motion techniques can provide an estimate of the geometry, this method returns, in addition to the geometry, the topological information in the form of a 1D path.

information from many images that yields correct, consistent and complete 3D reconstruction is a challenging problem. The second contribution introduces a physics based simulation approach to ensure the fidelity of the final result.

Chapter 2

Related Work

The problem of reconstructing a surface from points, images, volumetric data, etc., has received major attention in graphics and vision in the past two decades. Most popular methods fall into one of the following main categories: structure from motion (Wu et al. [1], Wu [2]), structured light (Valkenburg and McIvor [8]) and volumetric fusion (Curless and Levoy [9]). While these methods are general and able to produce high-fidelity 3D surfaces, they are challenged when they are used to reconstruct thin features with occlusions. This is due to the lack of a sufficient sampling of the thin structures, making it difficult to reconstruct a topologically correct surface. For instance, in KinectFusion (Izadi et al. [4]) and its derivatives, e.g., (Zhou and Koltun [3]), the quality of the reconstructed objects depends on depth map quality, in terms of image resolution and quantization of the error pattern.

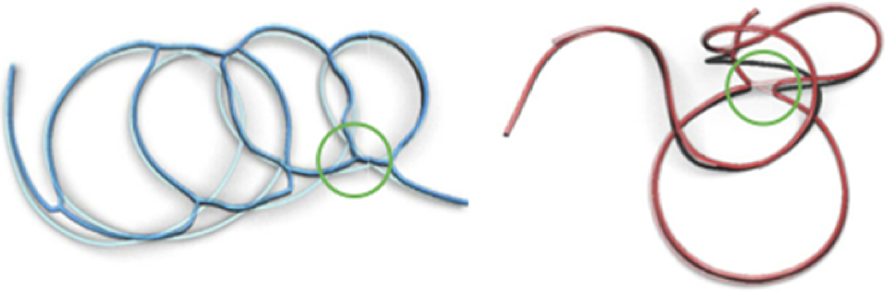


FIGURE 2.1: (Huang et al. [7]) medial axes found in the same datasets used for this thesis.

An additional challenge is that tubular structures often consist of self-occlusions, as the sensor cannot acquire images from every view on the object. This makes reconstruction of a watertight object difficult. Because of these challenges, application specific methods have been proposed. For instance, (Hu et al. [10]) propose a system to reconstruct strands of hair from an input point set acquired from a multi-view stereo setup. The method proposed in (Livny et al. [11]) employs a series of global optimizations to consolidate a point cloud representing one or more tree objects into skeletal structures, and uses a graph-based approach to reconstruct tree branches. However, tree structures have a simpler topology compared to more general graphs, thus, their optimization requires a relatively dense set of points. Similarly, the method of (Tagliasacchi et al. [6]) and (Huang et al. [7]) extract the medial axis from a point cloud and uses it to reconstruct the surface in a topologically correct way as shown in figure 2.1. (Li et al. [5]) introduce a new 1D primitive, called arterial snake, that is used to reconstruct 1D structures such as rods. Unfortunately, all these methods require a relatively dense set of sample points, much denser than can be obtained from one moving camera using state-of-the-art structure from motion algorithms, as demonstrated in the experiments section.

In the case of physics simulation systems, (Bergou et al. [12]) presents a discrete elastic rods formulation that allows curves to be simulated with rod properties (section 2.5).

The following sections give an overview on how current state-of-the-art general reconstruction methods are implemented (Sections 2.2 and 2.3). The result of these methods is always a point cloud. They all use a pin-hole camera model (section 2.1) to estimate the depth information of the images in 3D space.

Finally, the point cloud given by the general reconstruction methods is processed into a consistent representation of the object. Methods to obtain such representations are overviewed in section 2.4.

2.1 The camera model

There are two types of camera parameters used in computer vision for surface reconstruction: Intrinsic and extrinsic.

The intrinsic parameters form the perspective transformation of the camera which is defined by its focal length (the distance between the camera eye and the projection plane) and the physical coordinates of the projection plane (fig. 2.2). It can be observed, that if a camera has a short focal length, the field of view of the camera increases, and if the camera has a long focal length, its field of view narrows.

The intrinsic camera matrix is given in equation 2.1, where f is the focal length and ic is the image centre in pixels.

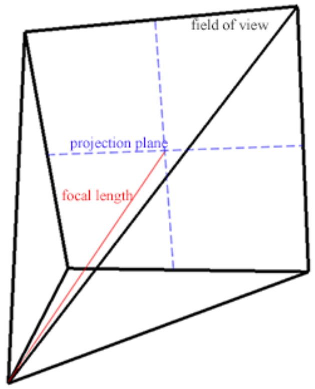


FIGURE 2.2: Intrinsic parameters of the camera.

$$K = \begin{pmatrix} f & 0 & ic_x \\ 0 & f & ic_y \\ 0 & 0 & 1 \end{pmatrix} \tag{2.1}$$

The extrinsic camera parameters determine its position and orientation in world coordinates. In reconstruction systems, there are always multiple cameras and usually the first one is placed at the origin. The other ones, are placed with respect to the first one. The extrinsic camera matrix is given in equation 2.2, where R is the rotation component and C is the camera position.

$$[RT] = \left[\begin{pmatrix} R_{1,1} & R_{1,2} & R_{1,3} \\ R_{2,1} & R_{2,2} & R_{2,3} \\ R_{3,1} & R_{3,2} & R_{3,3} \end{pmatrix} x + \begin{pmatrix} -C_x \\ -C_y \\ -C_z \end{pmatrix} \right] \tag{2.2}$$

2.2 Reconstruction from video

2.2.1 Reconstruction from calibrated setups

In calibrated setups, the parameters of the cameras are calibrated before performing the reconstruction. This calibration is done by moving a known pattern (eg. a chessboard, with known square width and height) (fig. 2.3) in front of the cameras, taking representative snapshots, and then tracing the pattern in the snapshots.

The intrinsic parameters are estimated individually. The pattern must be placed at different angles to increase the accuracy of the estimated intrinsic parameters of each camera.

The extrinsic parameters are estimated by guessing the position of the camera using the position of the pattern as reference, and then optimizing through iterative methods.

The main advantage of this method is that it is very accurate with low radial distortion. The main disadvantage is that the positions of the cameras must be fixed, otherwise the cameras have to be recalibrated each time they are moved.

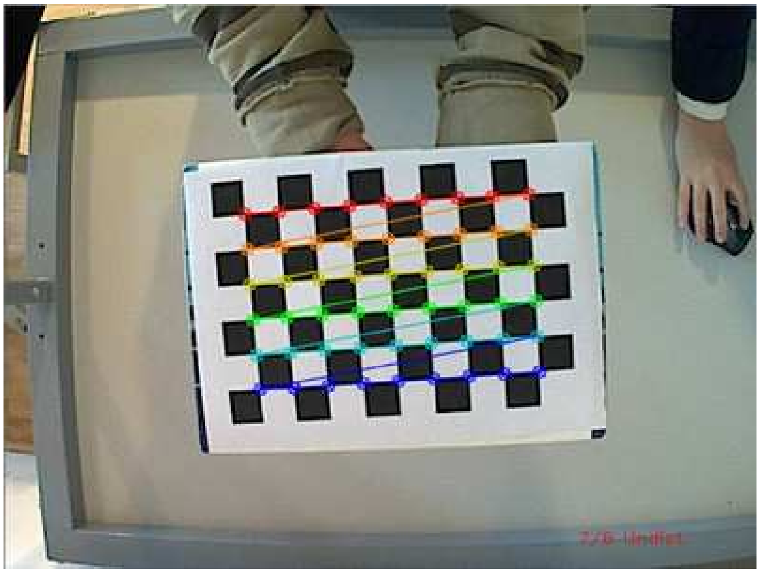


Figure 2.3: Chessboard pattern features.

Stereo reconstruction is a 3D reconstruction method that uses two cameras. With only one camera, it is impossible to estimate the depth of a point in a 3D scene. With two cameras, it is possible to estimate a point in a 3D scene by triangulating its position (fig: 2.4), thus making two cameras the minimum number required to perform a 3D reconstruction.

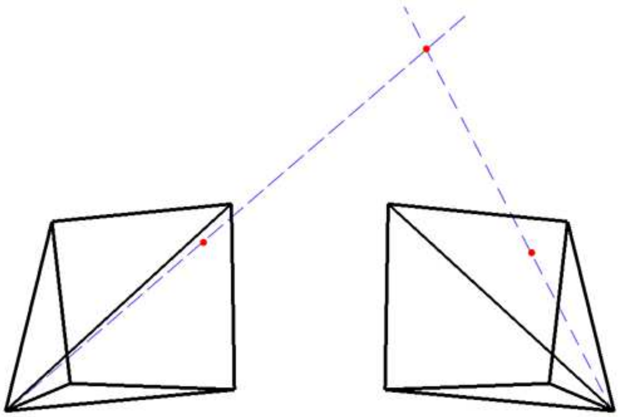


Figure 2.4: Two cameras triangulation.

The extrinsic and intrinsic parameters of the cameras must be known before performing the reconstruction, however this information is not enough as the positions of the point in image space (fig. 2.5) are also needed to be able to estimate its position in 3D. To find these common positions, several matching methods have been proposed such as Semiglobal Matching (Hirschmuller [13]).

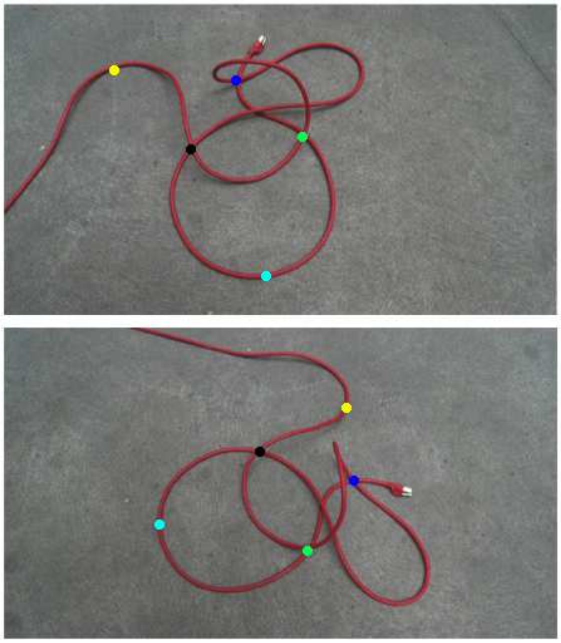


Figure 2.5: Common features in image space.

Modern techniques of multi-view stereo decompose the images into clusters to find the matches (Clustering Views for Multi-view Stereo) (Furukawa et al. [14]) and then reconstruct the scene by patches (Patch-based Multi-view Stereo) (Furukawa and Ponce

[15]). Improvements have been done to improve this kind of reconstruction in specific problems such as limited stereo pairs (Li et al. [16]).

The scene is finally reconstructed by identifying a good number of matches and triangulating their position in the 3D scene using the camera parameters.

The biggest limitation with stereo reconstruction is its poor ability to reconstruct objects with occlusions. To deal with this problem, this method can also be extended to support additional cameras to increase the accuracy of the reconstruction. This is known as multi-view stereo.

2.2.2 Reconstruction from uncalibrated setups

In uncalibrated systems, the parameters of the cameras are computed on the fly. There is no known pattern in the scene to follow, so the system must find a way to create these patterns instead. Once these patterns have been found, the camera estimation is done in a similar fashion. Contributions have been made to increase the number of patterns (Furukawa and Ponce [17]).

The biggest limitation of uncalibrated systems is that the patterns found may not be good and cause radial distortion. The main advantage is that it allows flexibility in the movements of the cameras.

Structure from motion (Wu et al. [1], Wu [2]) tries to reconstruct the geometry of a scene and estimate camera motion from a set of images.

Unlike other techniques exposed in this document, the cameras are not calibrated before the reconstruction. The cameras are estimated using only the information found in the images. With this in mind, Structure from motion proposes the pipeline illustrated in figure 2.6.

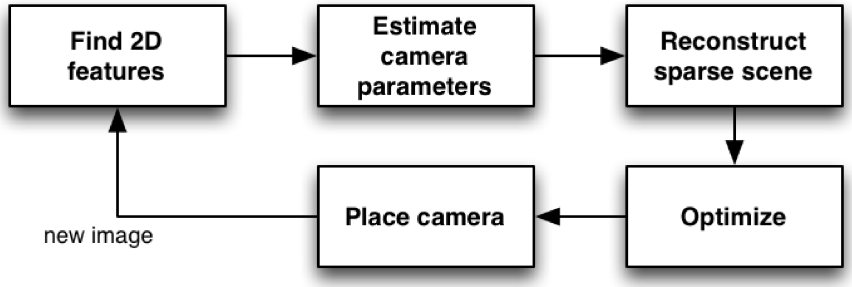


FIGURE 2.6: Structure from motion pipeline.

This method starts with two images. A first camera, for one of the two images, is placed at the origin. A set of features is found for each image and are corresponded. With these features, the position of the other camera is estimated with respect to the other. These two cameras are used to perform an initial reconstruction of the scene.

The next step is adding the remaining images to the scene. A set of features is found for all the images. Using the information of the cameras already placed, a new camera position is estimated.

With this new camera, the sparse scene is reconstructed again. This has the objective of identifying distortions and inconsistencies in the scene. These distortions are minimized by optimizing the cameras previously positioned in a process called bundle adjustment. Finally, the new camera is placed.

After all cameras have been estimated, it proceeds to perform the dense reconstruction of the scene.

This method is able to provide good reconstruction results with little additional information. However, it may suffer from considerable radial distortion and performance issues as bundle adjustment is an expensive routine.

Generally Structure from motion is followed by a dense reconstruction of the scene using PMVS (Furukawa et al. [14]), however the information given by SfM can be used to generate a surface directly from the sparse data such as manifolds (Lhuillier and Yu [18]).

2.3 Hybrid reconstruction

2.3.1 Structured light

Structured light (Valkenburg and McIvor [8]) is a 3D reconstruction method that uses a camera and a projector.

This method has a similar layout to stereo reconstruction, which has two cameras. In this case, one of the cameras is replaced by a projector for which the camera parameters still need to be known.

The aim of replacing one of the two cameras by the projector is to reduce the complexity in finding matches in the images. The projector casts a series of light stripes on the object (fig. 2.7) and now the matches become the light stripes. However, the position of these light stripes still has to be estimated. (Valkenburg and McIvor [8]) introduced a new method for estimating the 3D position of the sub-stripes of the light projected on the image using differential methods. Other methods present robust pixel classification [19] and calibration systems (Ben-Hamadou et al. [20]).

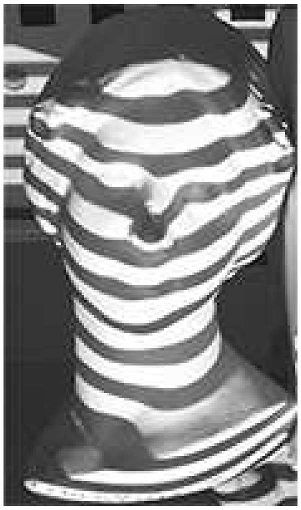


Figure 2.7: Light stripes on the object.

In terms of limitations, it shares the same limitation to stereo reconstruction, as its only change from it, is the way the features are found in the images.

2.3.2 Volumetric fusion

Volumetric fusion (Curless and Levoy [9]) reconstructs a 3D object using a range sensor and a laser projector.

In the previous kinds of reconstructions exposed, the depth of the 3D points was estimated by triangulating its position in image space. In this type of reconstruction, the RGB images are replaced by depth maps (fig. 2.8), so the depth is already given. Techniques that use silhouettes and range data have also been proposed (Yemez and Wetherilt [21]), (Song et al. [22]) .

The way the scene is reconstructed is implicit. After a series of depth maps have been taken, a volumetric grid is built (fig. 2.9). Each voxel is projected against the depth maps and it is turned on or off depending on the range data. (Curless and Levoy [9]) provide a way to join this range data. Notice that explicit reconstructions such as stereo and structured light directly compute an arbitrary number of points which are known to be part of the object. In a volumetric grid, the points are set beforehand and later it is decided if it belongs to the object or not.

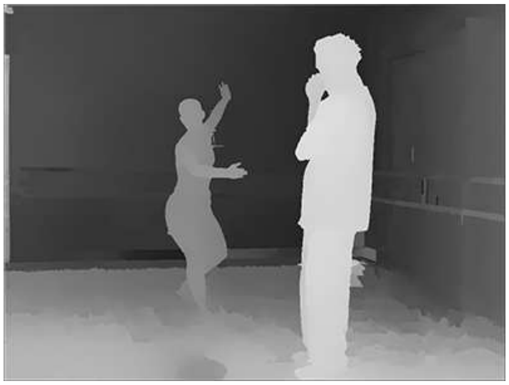


Figure 2.8: Depth map. Objects closer to the sensor are whiter.

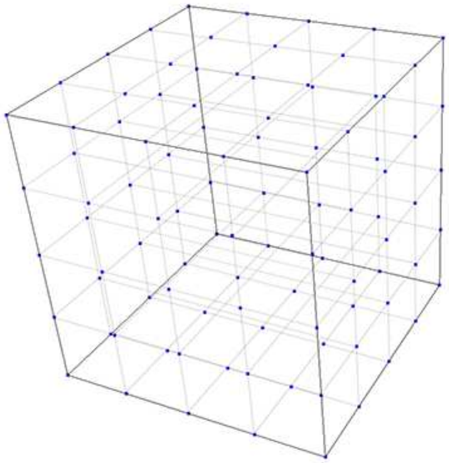


Figure 2.9: Volumetric grid.

The main advantage of this method is its performance. Since it is no longer needed to find matches in the images, and the depth map is given directly, this method is very fast. The main disadvantage is that reconstructing thin objects is difficult due to hardware limitations.

2.4 Object representation

In the previous sections, several methods were exposed on how to get a set of points belonging to an object from a video or hybrid setups. This set of points, however, is a raw representation of the object being reconstructed and can be manipulated depending on which kind of representation is desired.

Point based surfaces introduce new surface estimation mechanisms to allow point clouds to be correctly rendered without finding a triangular mesh. Surface meshes methods try to get a triangular mesh from the set of points. Volume meshes are a representation of the interior of the meshes. 1D reconstruction techniques aim to fit a set of 1D curves through the point set.

2.4.1 Point based surfaces - MLS

The main idea behind point based surfaces is that common methods of reconstruction provide a relatively good point cloud as an initial model, so they can be used to be a good representative of the physical model. The main problem with this initial point cloud is that it is not smooth, and outliers are very common. Point based surfaces and MLS methods (Alexa et al. [23]) aim to provide a smooth surface estimation technique which preserves small details of the object being reconstructed (fig. 2.10).

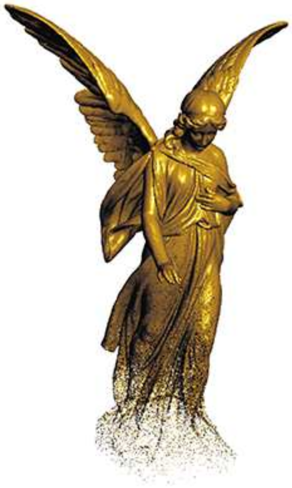


Figure 2.10: Point based surface.

MLS estimates a polynomial that fits the set of points using moving least squares optimization (Lee [24]). The points are resampled using the polynomial and a local plane computed for each point (fig. 2.11).

In figure 2.11, g corresponds to the approximated polynomial, r to the original point, H is the reference plane of the original point, and the point between r and q defines the resampled point. This way, the points are smoothed and are allowed to be rendered without noise.

This method preserves better the structure of the object than a triangular mesh, however it may suffer if the point cloud is not very clean.

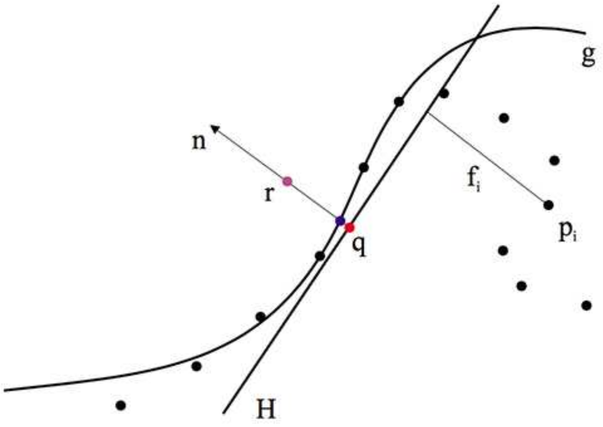


FIGURE 2.11: Surface resampling.

There have been contributions to reconstruct thin object using Point based surfaces methods by forcing points into a common surface (Ummenhofer and Brox [25]).

2.4.2 Surface mesh

Surface extraction methods aim to find a triangular mesh out of a point cloud. The motivation behind this representation is that triangular meshes is the most common type of representation in existence, and usually the object extracted will most likely be used in a system where a triangular mesh is needed.

This extraction is done in different ways such as, Delaunay triangulation (Kuo and Yau [26]), the Voronoi diagram of the point cloud (Alliez et al. [27]) or the most common one, Poisson surface reconstruction (fig. 2.12) (Kazhdan et al. [28]).

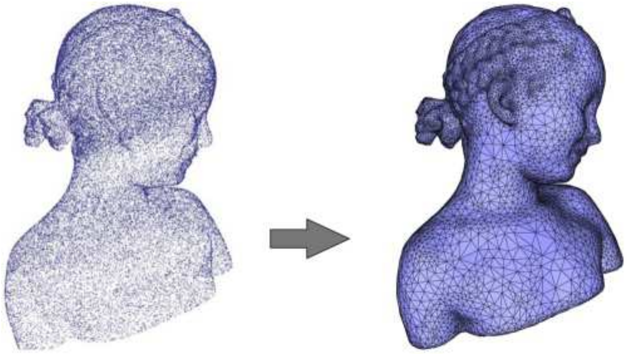


FIGURE 2.12: Poisson surface reconstruction.

Methods based on Delaunay triangulation and Voronoi diagrams of the point cloud create jagged meshes if the points are subject to noise. Poisson surface reconstruction finds an approximate surface to the points. None of these methods is good to reconstruct thin structures since their point cloud is often sparse and these methods rely on a dense point cloud.

2.4.3 Volume mesh

Volume meshes are used when the information given by surface meshes is not enough, and information about the interior of the mesh is desired.

The simplest method used to obtain a volume mesh is resampling a surface mesh. Several vertices are added in the interior of the mesh and are connected using Delaunay triangulation (fig. 2.13 or similar. This kind of mesh is used in volumetric deformation methods such as the one proposed by (Zhou et al. [29]).

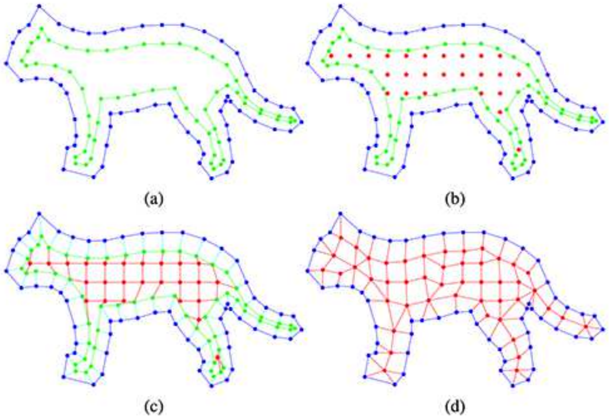


Figure 2.13: Volumetric graph construction.

Volumes are also used for finding a triangular mesh out of implicit functions with the Marching Cubes method (Nielson [30]).

In reconstruction, volumes are used to reconstruct a scene implicitly as shown in section 2.3.2. This kind of models offers a good intermediate representation for reconstructing thin structures.

2.4.4 1D reconstruction

This kind of reconstruction consists in a poly-line or a set of poly-lines. They are mainly used for analysis of the point cloud and to extract structural or topological features from it.

Arterial snakes (Li et al. [5]) tries to reconstruct thin tubular structures in a point cloud by growing "snakes" from such tubular structures (fig. 2.14). It aims to provide an accurate physical representation of the object scanned by using only poly-lines.

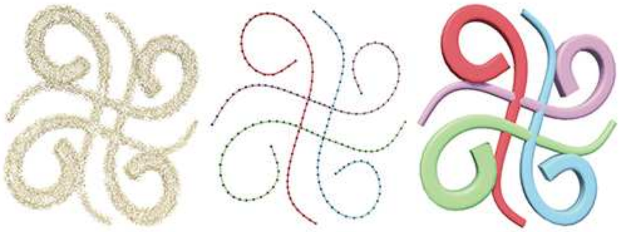


FIGURE 2.14: Arterial snakes.

L1-medial skeleton (Huang et al. [7]) extracts a skeleton that corresponds to the median of the raw scan (fig. 2.15). This skeleton is a powerful analysis tool and representation of the model. It is not meant to be an accurate representation of the physical model.

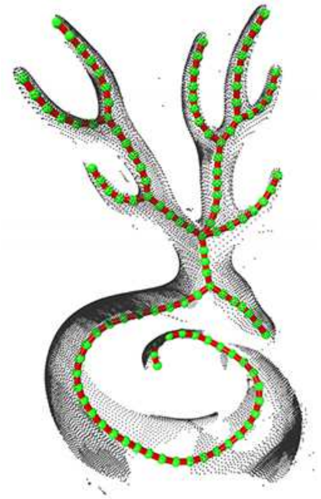


FIGURE 2.15: L1-medial skeleton.

2.5 Physics (Discrete elastic rods)

Discrete elastic rods (Bergou et al. [12]) proposes a discrete geometric model for Kirchhoff rods. Its main objective is to introduce a simple and easier to implement model than existing ones.

Kirchhoff rods defines the concept of an adapted framed curve 2.16 to represent bending and twisting motion. A rod consists in a centreline and orthonormal frames on every point $\{t(s), m_1(s), m_2(s)\}$. These frames are then used to compute the bending and twisting energy in each frame (eq. 2.3).

$$\omega_1 = t' \cdot m_1 \tag{2.3a}$$

$$\omega_2 = t' \cdot m_2 \tag{2.3b}$$

$$m = m_1' \cdot m_2 \tag{2.3c}$$

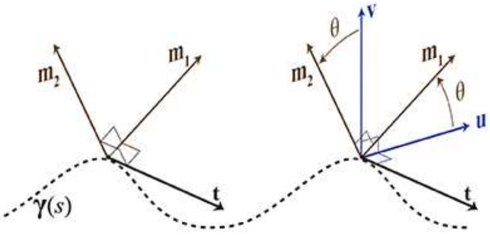


FIGURE 2.16: Adapted frame curve.

ω_1 and ω_2 represent the bending energy, which is the curvature at that point in the curve. m represents the twisting energy which is the rotation around the centreline of the rod. Using this model, the Kirchhoffs continuous bending and twisting energies for the whole rod (isotropic) are pictured in equations 2.4 and 2.5. α and β correspond to the bending and twisting modulus, which define the stiffness of the rod.

$$E_{bend}(\Gamma) = \frac{1}{2} \int \alpha \boldsymbol{\omega}^2 ds \tag{2.4}$$

$$E_{twist}(\Gamma) = \frac{1}{2} \int \beta m^2 ds \tag{2.5}$$

The overall energy is defined in equation 2.6.

$$E(\Gamma) = E_{bend} + E_{twist} \tag{2.6}$$

In the case of the bending energy $\boldsymbol{\omega} = (w_1, w_2)^T$.

(Bergou et al. [12]) contributed with the formulation of the discrete energies pictured in equations 2.7 and 2.8.

$$E_{bend}(\Gamma) = \frac{1}{2}\sum_{i=1}^{n} \alpha \left(\frac{\kappa \boldsymbol{b}_i}{\bar{l}_i/2}\right)^2 \frac{\bar{l}_i}{2} = \sum_{i=1}^{n} \frac{\alpha(\kappa \boldsymbol{b}_i)^2}{\bar{l}_i} \tag{2.7}$$

$$E_{twist}(\Gamma) = \sum_{i=1}^{n} \beta \frac{(\theta^i - \theta^{i-1})^2}{\bar{l}_i} = \sum_{i=1}^{n} \frac{\beta m_i^2}{\bar{l}_i} \tag{2.8}$$

In the bending energy, $(\kappa \boldsymbol{b})_i$ is the curvature binormal and is defined by equation 2.9, where $\boldsymbol{e}^i = \boldsymbol{x}_{i+1} - \boldsymbol{x}_i$, being $\boldsymbol{x}_i$ the ieth vertex of the rod. $\bar{l}_i$ is double the distance of the Voronoi region of the vertex, and is defined as $\bar{l}_i = \|\boldsymbol{e}^{i-1}\| + \|\boldsymbol{e}^i\|$.

$$(\kappa \boldsymbol{b})_i = \frac{2\boldsymbol{e}^{i-1} \times \boldsymbol{e}^i}{\|\bar{\boldsymbol{e}}^{i-1}\|\|\bar{\boldsymbol{e}}^i\| + \boldsymbol{e}^{i-1} \cdot \boldsymbol{e}^i} \tag{2.9}$$

With the discrete energies defined, the next step is to perform the physics simulation. The initial rod has its material frames computed for each vertex with no twist. After an external force is applied, the objective is to minimize the total energy of the system. This is done through numerical methods.

Chapter 3

Proposed Approach

The method proposed here combines uncalibrated multi-view stereo from Structure from Motion, a volumetric representation of the surface and rods physics to reconstruct cable-like objects.

There are several challenges when trying to reconstruct a tubular structure with multiple crossings. The first one is detecting those crossings. In 2D, a simple skeletonization algorithm can solve the problem. However, it is not so clear in 3D scenarios. Another challenge is processing the volumetric grid obtained in the initial step of a reconstruction. In the case of thin tubular structures, this volumetric grid may have some wedges and present some minor inconsistencies due to distortion error during the camera calibration step. It is also hard to find the medial axis of a volumetric grid. The final challenge is achieving a smooth cable-like looking curve. The medial axis of a mesh is not usually smooth and follows the topology of the whole grid rather than the topology of the object being reconstructed.

With this challenges in mind, the method takes as input a sequence of colour images of a tubular structure. The reconstruction pipeline consists of the following steps:

1. Segment the tubular object pixels and identify the junction regions in the 2D images (Section 3.1).
2. Fuse images and junction information from the 2D images into a 3D occupancy grid (Section 3.2).
3. Estimate 1D curve skeleton segments that connect junction regions or end-points in 3D (Section 3.3).
4. Reconstruct the geometry of the tube segments combining the information from the video footage with rod physics (Section 3.5).

5. Join the 1D segments preserving the topology of the cable (Section 3.6).

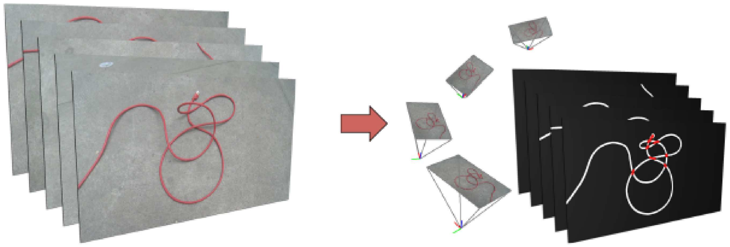


Figure 3.1: Camera estimation and segmentation overview.

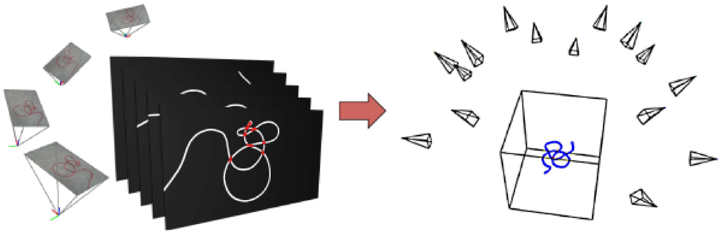


Figure 3.2: Volumetric grid computation overview.

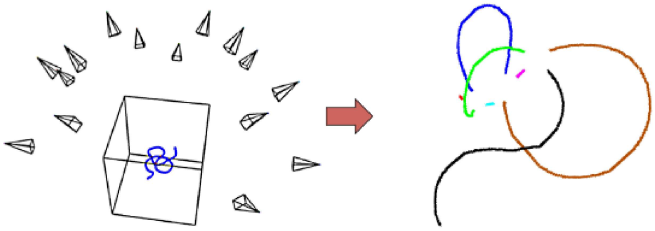


Figure 3.3: Paths extraction overview.

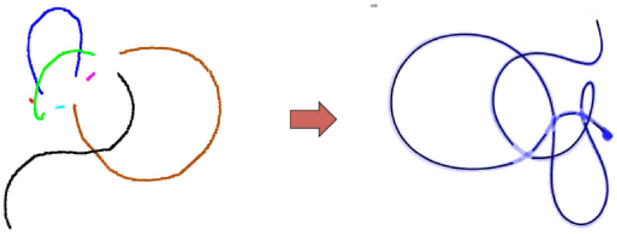


Figure 3.4: Physics simulation overview.

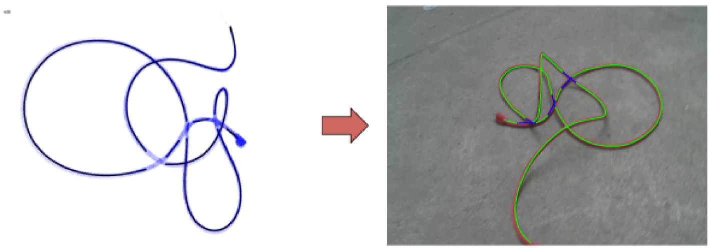


Figure 3.5: Path unification overview.

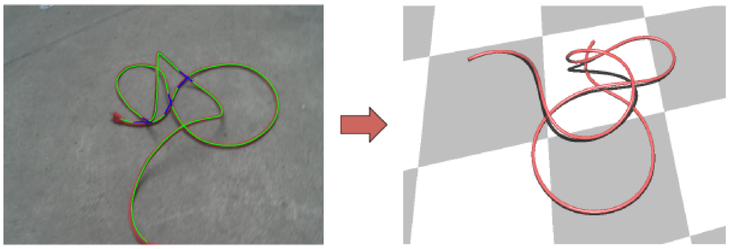


Figure 3.6: Result overview.

3.1 2D Segmentation and Junction Regions

Each image is converted to HSV color space and then thresholded for one frame of the sequence with an intuitive slider. All the other images of the sequence are then automatically processed. Pixels corresponding to the tubular object are referred as tube pixels and are marked white, otherwise as non-tube pixels and marked black. Junction regions are collections of pixels where the cable overlaps with itself. Intuitively, these regions contain more tube than non-tube pixels. Hence, these regions are detected by making the assumption that a pixel belongs to a junction region when the number of tube pixels in a local window around this pixel is larger than in a window of a pixel away from the junction region. If that number exceeds a threshold, the pixel is considered a junction pixel. While this is a relatively simple approach, it is efficient to compute and works well for all the examples shown in this document. The result is shown in figure 3.7.

The threshold by which a pixel is considered to be a junction varies by dataset. However, the same threshold applies to all of the images being processed within the same dataset. It has to be noted that the junctions may not be correctly marked in all of the images as shown in figure 3.8, but the amount of images where the junctions are correctly marked are more significant than the number of images where it is not, and it is possible to get a good occupancy grid to work with in the following steps for most cases.

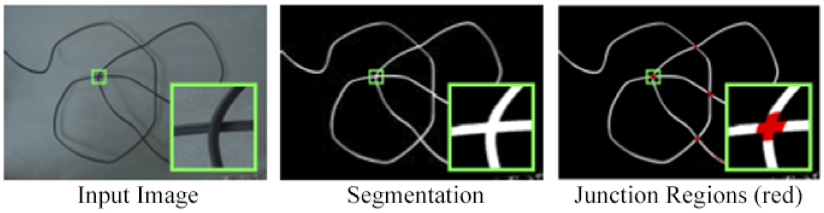


Figure 3.7: Segmentation of the input image (left) into tube and non-tube pixels (middle), and identification of the junction regions.

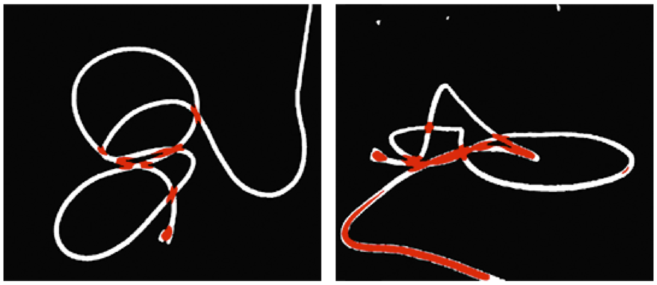


Figure 3.8: Result for the junction finding algorithm in two images of the same dataset.

3.2 3D Occupancy Grid Construction

VisualSFM [1, 2] is applied, based on structure form motion, to the input video sequence to estimate the intrinsic and extrinsic parameters of the cameras. The information from all the images is fused, by embedding the scene into a regular 3D grid placed in the area of interest. Grid voxels are labeled as inside the cable and marked as occupied if they project into the segmented areas in at least half of the images. Same is applied to junction cells. Figure 3.9 illustrates the grid resulting from this classification.

The camera information from VisualSFM is translated to its OpenGL homologue for ease of use. So for a given K[R T] (eq. 3.1), the projection and modelview matrices are computed (eq. 3.2).

$$K[RT] = \begin{pmatrix} f & 0 & ic_x \\ 0 & f & ic_y \\ 0 & 0 & 1 \end{pmatrix} \left[\begin{pmatrix} R_{1,1} & R_{1,2} & R_{1,3} \\ R_{2,1} & R_{2,2} & R_{2,3} \\ R_{3,1} & R_{3,2} & R_{3,3} \end{pmatrix} x + \begin{pmatrix} -C_x \\ -C_y \\ -C_z \end{pmatrix} \right] \tag{3.1}$$

In equation 3.2 (t,b,l,r) correspond to the top, bottom, left and right coordinates in world space of the screen where ic is the centre of the image, and (n,f) correspond to the near and far values. p^{tr} and p^{bl} correspond to the top-right and bottom-left points of the screen, K is the projection matrix, and p_x, p_y and p_z correspond to the x,y,z coordinates of the point, so p_y^{tr} is the y coordinate of the top-right point.

$$PM = \begin{pmatrix} 2\frac{n}{r-l} & 0 & -\frac{r+l}{r-l} & 0 \\ 0 & -2\frac{n}{t-b} & -\frac{t+b}{t-b} & 0 \\ 0 & 0 & \frac{f+n}{f-n} & -2\frac{nf}{f-n} \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} & & & -C_x \\ & R & & -C_y \\ & & & -C_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{3.2}$$

Projection matrix parameters :

$$p^{tr} = K^{-1} \begin{pmatrix} ic_x & ic_y & 1 \end{pmatrix}, \tag{3.3a}$$

$$p^{bl} = K^{-1} \begin{pmatrix} 0 & 0 & 1 \end{pmatrix}, \tag{3.3b}$$

$$t = n\frac{p_y^{tr}}{p_z^{tr}}, b = n\frac{p_y^{bl}}{p_z^{bl}}, \tag{3.3c}$$

$$l = n\frac{p_x^{bl}}{p_z^{bl}}, r = n\frac{p_x^{tr}}{p_z^{tr}} \tag{3.3d}$$

This occupancy grid is treated as a connected graph where the nodes are the corners of the turned on voxels, and the edges are the paths between two consecutive turned on voxel corners. The outliers are removed leaving only one graph. The nodes are classified as either simple nodes or junction nodes. Figure 3.10

3.3 3D Segments Extraction

Although the 3D occupancy grid provides some rough information regarding the location and geometry of the tubular structure, it has no knowledge of its topology. This method extracts 1D curves that connect two junctions or a junction to an end-point (segments) by traversing through the occupied regions of the grid. Although junctions are identified, extracting the segments is non-trivial because there can be several segments joining the

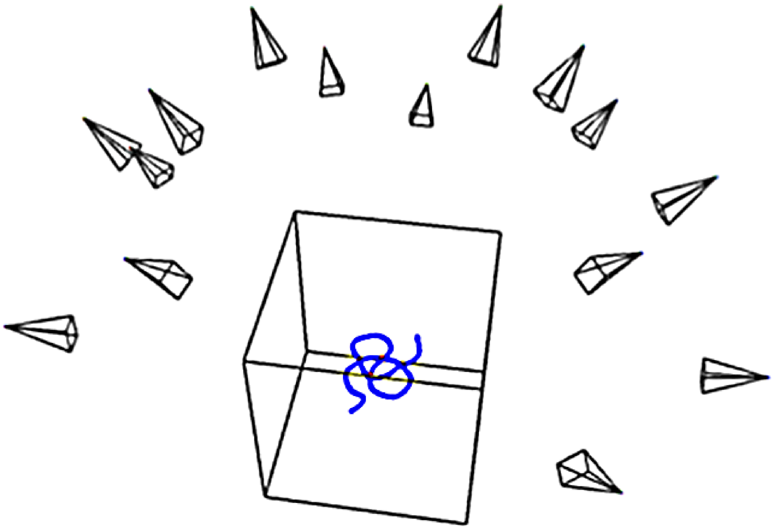


Figure 3.9: Images from the cameras are embedded into a 3D volumetric grid (only a subset of the cameras are shown). Voxels marked in blue are occupied by the tube structure.

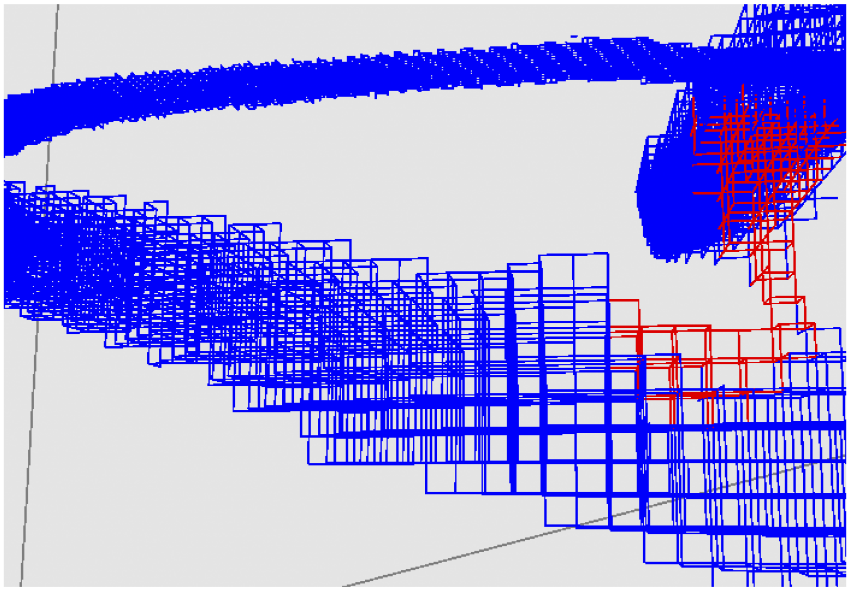
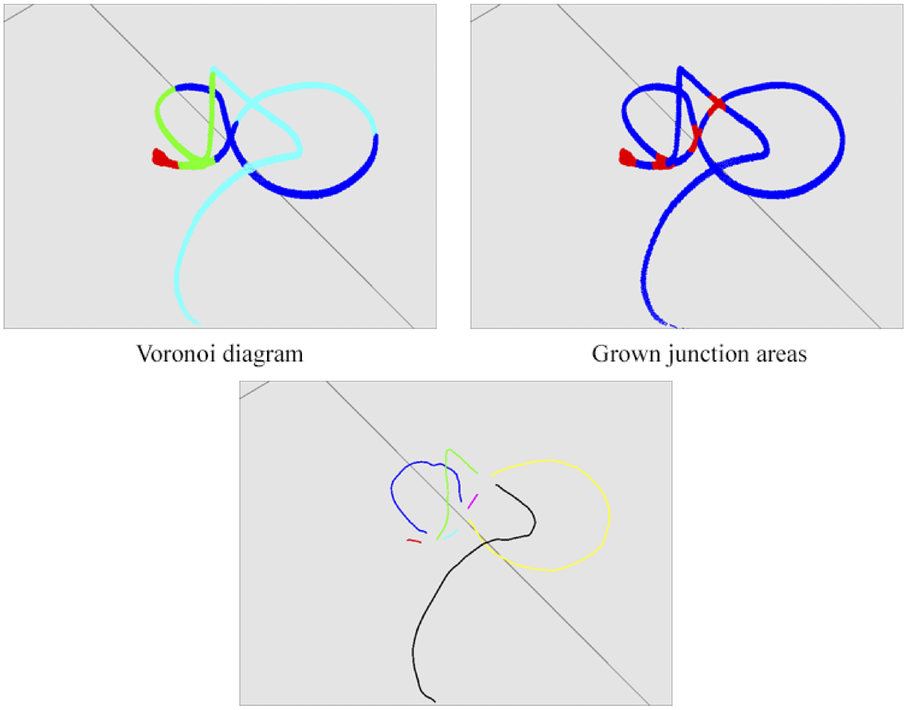


Figure 3.10: Close up to the graph formed by the voxels.

same two junctions, or there could be loops. To solve this problem, it is important not to only identify the junction regions, but also to determine how many segments emanate from a given junction.

This process is done with the following pipeline.



Voronoi diagram

Grown junction areas

Initial Guess

Figure 3.11: Illustrations of the Voronoi diagram, grown junction areas and found paths.

Identify all the junction regions from the raw graph. All the junction nodes are traversed and classified. If a junction node has not been classified yet, a local search is performed and every junction node connected through junction nodes is marked to belong to the same junction area.

The centroid of each junction area is chosen by selecting the junction node closest to the average of the nodes in the area. This node will be referred to as the central junction node.

The Voronoi diagram of the graph is found by growing all the central junction nodes using the Breadth First Search algorithm one level at a time (see section 3.4) sequentially, until they meet each other.

Each central junction node is grown using Breadth First Search until a maximum number of fronts is found per junction area, being the Voronoi diagram (fig. 3.11) the limit. Every node traversed by the BFS algorithm is marked as a junction node in order to obtain a more consistent junction area (fig. 3.11). A front is a subgraph of simple nodes that have at least one edge connecting to a junction node from the newly computed junction areas (see section 3.4).

The centroid of each front is found the same way the central junction nodes are found.

A path is grown from each front until it finds another front. If it does not find another front, the longest path is chosen instead. These paths become the initial guess of the tube (fig. 3.11).

The paths are smoothed using the Laplacian to make them suitable for the physics simulation. In equation 3.4 p' corresponds to the smoothed vertex, N to the number of neighbours the vertex has, which in this case is 2 and p_j is the position of the neighbouring vertex.

$$p' = \frac{1}{N}\sum_{i=1}^{N} p_j \tag{3.4}$$

3.4 Growing Algorithm

The junction area growing procedure is based on the Depth First Search algorithm. Consider the figure 3.12. The process starts with one node. Each level corresponds to the neighbours of the nodes already marked. To advance one level, the neighbours of the nodes are marked as junctions.

To identify the fronts, consider the nodes marked in green. A front consists of consecutively connected green nodes by up to one red node (fig. 3.13). This method is guaranteed to work thanks to the way the junctions were grown. The green nodes of a single front are always guaranteed to be connected by at least one red node.

With these fronts, it is possible to proceed to connect the paths with the method exposed in section 3.3, and there is no space between two fronts where the cable guess might escape while growing to find a path between them. The front can be seen as a wall dividing the graph.

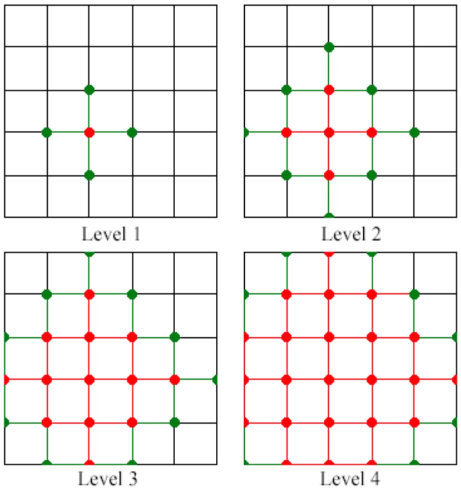


FIGURE 3.12: Growing algorithm. State at different levels.

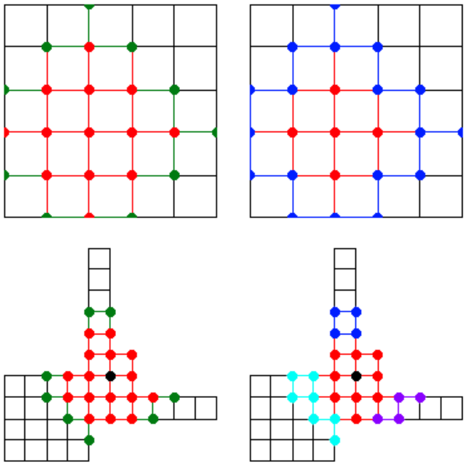


FIGURE 3.13: Fronts identification. Left: Graph at a current state. Right: Fronts identified. Each independent front is in a different colour.

3.5 3D Segments Reconstruction

While the segments identified in the previous step provide accurate topological information regarding the tube, they are generally not geometrically accurate. Accuracy is improved by executing a physics based rod simulation based on (Bergou et al. [12]). The external forces of the simulation are computed based on the occupancy grid. Each vertex in the occupancy grid which is sufficiently close to the rod exerts a force onto the closest vertex of the rod. This has the effect of naturally placing the rod inside the occupied grid voxels while at the same time pressuring its physical properties. Figures 3.14, 3.15, 3.16, 3.17.

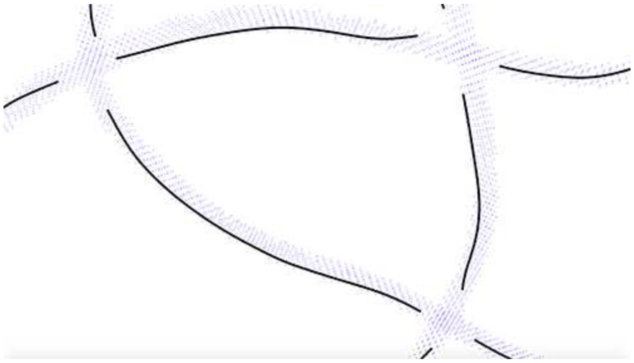


Figure 3.14: Segments and point cloud.

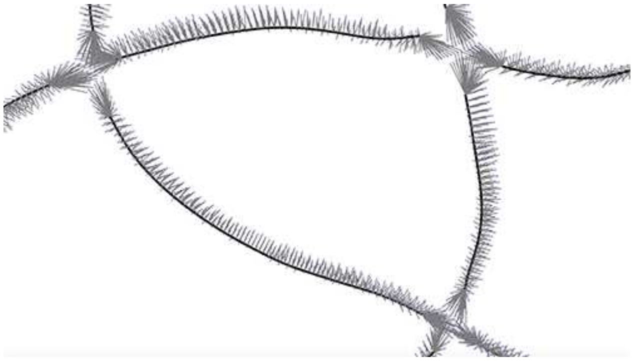


Figure 3.15: Computed correspondences. Each vertex in the point cloud exerts a force on its closest vertex in the segment.

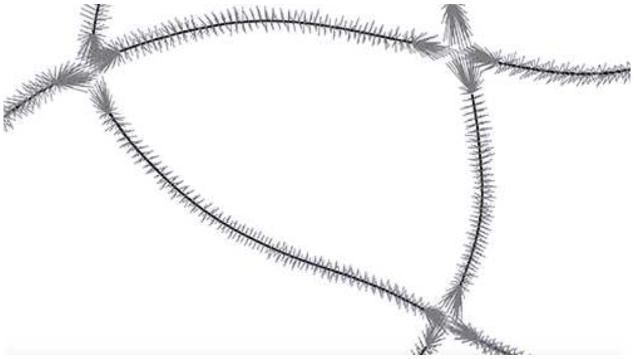


Figure 3.16: Correspondences after simulation.

The physics simulation makes use of the bending and twisting energies to maintain the physical properties of the rod with an inextensibility constraint. Mass damping is added to the system to make sure the system converges. The system is then solved using Newton-Rhapson method to find the velocities of the vertices at each time step.

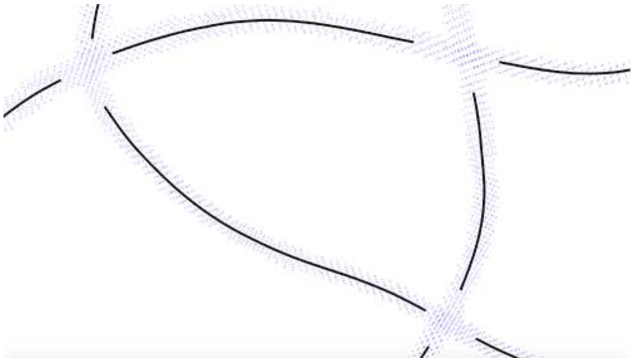


Figure 3.17: Segments after simulation.

One challenging problem is computing the external force. If the force is too weak, the physics simulation would be useless, but if the force is too high, the system becomes unstable. Hence, the external force in a vertex is computed by adding the normalized vectors from the vertex to the vertices in the point cloud that were chosen to be closest.

Equation 3.5 refers to the mass damping used for the simulation and equation 3.6 to the external force. Both are vectors.

$$F_{md} = -cm_i v_i \tag{3.5}$$

In equation 3.5 c corresponds to the magnitude, m to the mass, i to the vertex index and v to the velocity of the vertex.

$$F_e = cm_i \sum_{j=0}^{n} \frac{p_j - p_i}{\|p_j - p_i\|} \tag{3.6}$$

In equation 3.6 c corresponds to the magnitude, m to the mass, i to the vertex index, j to a vertex index correspondence in the point cloud and p to the position.

There are several parameters of the simulation that have to be carefully tweaked for each dataset, however this tweaks can be done automatically depending on the number of vertices in the point cloud, its density and how close together the vertices in the 1D curve are. These properties are explained in table 3.1.

The physics simulation is considered done once there are no significant changes from one time step to the next.

Property	Description
Density	This parameter defines the stiffness of the rod. If it's low, the rod is better preserved, however it becomes less manipulable.
Mass	The mass of each vertex can be left constant for all datasets.
Force	The magnitude of the force that has to be applied to each vertex.
Damp Force	The magnitude of the desacceleration force.

Table 3.1: Physical properties.
Properties of the physics simulation.

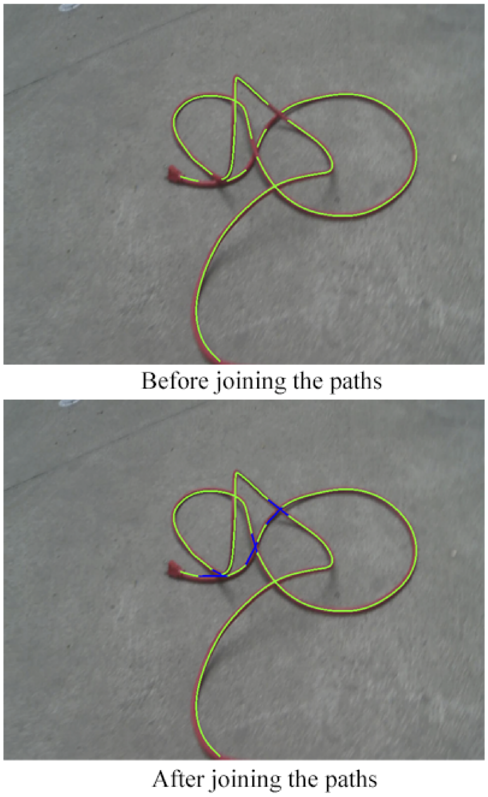
3.6 Topological 3D Segments Connection

The reconstructed tubular segments need to be connected into a single tube. Determining how segment end points are connected to other segment end points at the junctions regions is a combinatorial problem. Once again, visual information is used from the images to solve this. When cables cross, the cable that goes on top has no sharp edges while the cable at the bottom generally exhibit two sharp edges due to the ambient occlusion. Then, in image space, the shortest path between the end points using the image gradient as graph weights is computed. Figure 3.18.

The average of the shortest path through the gradient from the junction end points is computed for all the images and the segments are chosen from the shortest distance to the longest one by one until there are only two left, which are the tube end points. This way, the system will connect first the end points representing the top crossings of the cable and the bottom crossings are connected last. It is noticeable then that this method only works well when there are less than four junction end points per junction area, and solving more complex scenarios is left for future work. To improve performance, only paths between two junctions end points belonging to the same junction area are considered.

The gradient which gives the best result uses a 3x3 Gaussian filter kernel as seen in equation 3.7 and table 3.2. In equation 3.7 $G(x, y)$ corresponds to the 3x3 convolution matrix around the pixel (x, y).

$$g(x, y) = \|image(x, y) - G(x, y)\| \tag{3.7}$$



Before joining the paths

After joining the paths

Figure 3.18: Images with the paths before and after connection.

0.075	0.124	0.075
0.123	0.204	0.123
0.075	0.124	0.075

Table 3.2: Gaussian kernel.
A gaussian 3x3 kernel with covariance 1 normalized.

One additional problem is that tubular structures such as cables are often made out of plastic material that may exhibit specular highlights as shown. This can interfere with the computations as edges that cross such a highlight will still have a large weight. However, specular highlights tend to be aligned along the cable, therefore the shortest path is computed by choosing the brightest pixel in a small vicinity of the end points. This method works in most cases. However, in the rare case when it does not work, the system has a further combinatorial check. If the connections are made incorrectly, the result might be two tubular structures instead of one. This case is automatically detected by the system and the user is prompted for high level assistance.

A connection is regarded as ambiguous when the shortest paths between junctions in the same junction area are not significantly different.

Finally, once the connections are made, rod-based physics simulation presented in Section 3.5 is performed once again on the entire structure to obtain the final results. This last step is important as it smoothes out wriggles at the segments connection points.

3.7 Implementation

The system is constructed using the following components: VisualSFM (Wu et al. [1], Wu [2]), a volumetric grid application, a path extractor, a path unifier, and GIMP and Matlab scripts.

A video is given as the input. An image is taken from the video each 10 frames with the idea of getting a good calibration from VisualSFM. These frames are obtained using a MATLAB script.

The images obtained are processed with VisualSFM (Wu et al. [1], Wu [2]). The result of this process is a text file with the camera information of the images. This file is processed by a script written in C++ to a format compatible with the system.

The images are segmented in bulk using a GIMP script written in Python. Similarly, a GIMP python script is used to find the gradient, which is precomputed to save on time.

The volumetric grid application is written in C++. It receives as input the segmented images and the camera information. It outputs the resulting grid in two formats; as a graph with nodes and edges, and as a bit array with all the cells of the grid. The last format is used for visualization purposes.

The path extractor is also written in C++. It receives the graph as the input and outputs smoothed initial guesses.

The physics simulator (Bergou et al. [12]) in this step receives the volumetric grid and the smoothed initial guesses or final guess. It outputs the physically correct initial guesses or the final result.

The path unifier receives the segmented images, the gradient of the images and their camera parameters, and the output of the physics simulator. It outputs a single 1D path whose junctions are connected by a sampled straight line.

3.8 Constraints

This method has a number of constraints which will be reviewed in this section.

The input video must be recorded with the light placed ideally on top of the cable and remain constant through the whole video. If placed in a different manner, it will create undesirable shadows that would difficult the segmentation of the cable and cause the gradient algorithm to fail. If the light position changes abruptly during the video, a good calibration may not be able to be obtained.

The camera must be moved slowly around the cable, capturing as much detail as possible. The system must get good camera calibration parameters and a relatively clean point cloud. The formation of wedges in a 3D point cloud is often undesirable. The system can handle minor wedges but it is very sensitive with wedges over the junction areas. Wedges on junction areas will cause the growing algorithm to fail and unable to find correct junctions.

A wedge is a set of points in the point cloud that are considered to be part of the object when they should not (fig. 3.19). Usually, it is noise generated by camera distortion or occlusions in the images.

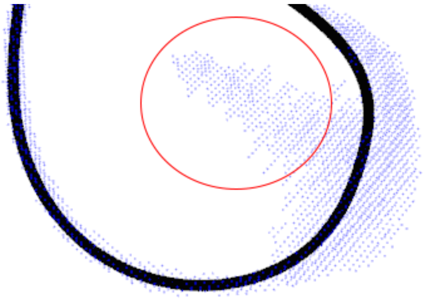


FIGURE 3.19: A wedge in the point cloud. The area circled in red is considered to be part of the cable even though that area does not belong to the cable. The guess of the cable is given in black

In terms of resolution, any modern phone camera can be used to get the video, however, the resolution should be over 1200x800 and should not have too much noise. Little noise can be solved by applying some blur to the image, however noise found in pictures taken at night, for example, or cameras with poor optics, will cause both the segmentation, and the gradient to fail.

The system can handle cables of multiple thicknesses, however it will most likely fail if the cable is too thin (e.g. earphones). It is very difficult to get a good graph/point cloud with a cable of these characteristics, the images would need to have a huge resolution and the grid would need to be very dense as well. The gradient may also fail in this case.

Cables with deformation properties may be reconstructed, but with some problems. The physics simulation system assumes a stiff rod. If the stiffness of the rod is reduced to allow this kind of cables, the physics simulation becomes useless as the elastic forces no longer have major impact in the modelling of the cable. The result will be jagged.

Chapter 4

Results and Discussion

In this section, implementation details of the system and the results obtained are described.

The system implemented uses VisualSFM (Wu et al. [1], Wu [2]) and a modified version of the physics simulation implementation done by (Bergou et al. [12]). All the code is written in C++ using OpenGL 2.1 for rendering, Eigen for math, and Glui for the user interface in a Linux environment. The details are discussed in the following section.

The accuracy of the method is tested using synthetic datasets and real datasets. The discussion will be focused in the evaluation of two synthetic datasets and three real datasets. All two synthetic datasets are manually created poly-lines which are then rendered from different views. The results are evaluated by difficulty in obtaining the occupancy grid, quality of the occupancy grid, quality of the junctions, quality of the images and quality of the end result.

In synthetic datasets it is possible to evaluate the algorithm quantitatively since the reconstructed cable should be the same as the synthetic cable. This accuracy is evaluated by calculating the distance from each vertex in the output polyline to the closest line segment in the input polyline. The error is the average of these distances.

Real datasets can only be compared qualitatively since developing a quantitative comparison method equals to solving the problem.

4.1 Synthetic datasets

All two synthetic datasets are evaluated under the same section, as the input is similar and there are not many changes among them when comparing its quality. The purpose of the synthetic datasets is to test the pipeline accuracy in simple cases before going to real datasets. The first synthetic dataset tests the algorithm on complex crossings, and the second evaluates the algorithm capacity on processing loops with 3D complexity. Thus, the synthetic datasets have the following characteristics:

1. The intrinsic and extrinsic camera parameters are perfect since they are obtained directly from the renderer.
2. The images exhibit clear junctions with clear shadows where the cable crossings are located.

With these characteristics, a clean occupancy grid is obtained for every dataset even at low grid resolutions with clear junctions. All two cases are solved using the same grid resolution and parameters for the physics simulation. Since the camera parameters are perfect, a voxel in the occupancy grid is considered to be on only if it is white in all of the segmented images. Same with the junctions. The results are summarized in table 4.1. Since the input is synthetic, it is possible to make a direct comparison with the output of the system. This is shown in figure 4.2.

Property	Result
Occupancy Grid	The occupancy grid is trivial to obtain, and free of wedges or noise.
Junctions	The junctions were manually computed for the synthetic datasets, so they are perfect.
the Images	The images exhibit clear shadows in the crossings.
End result	The difference between the synthetic dataset and the output is minimal. Quantitative results are given in section 4.5.

Table 4.1: Result for the synthetic datasets.

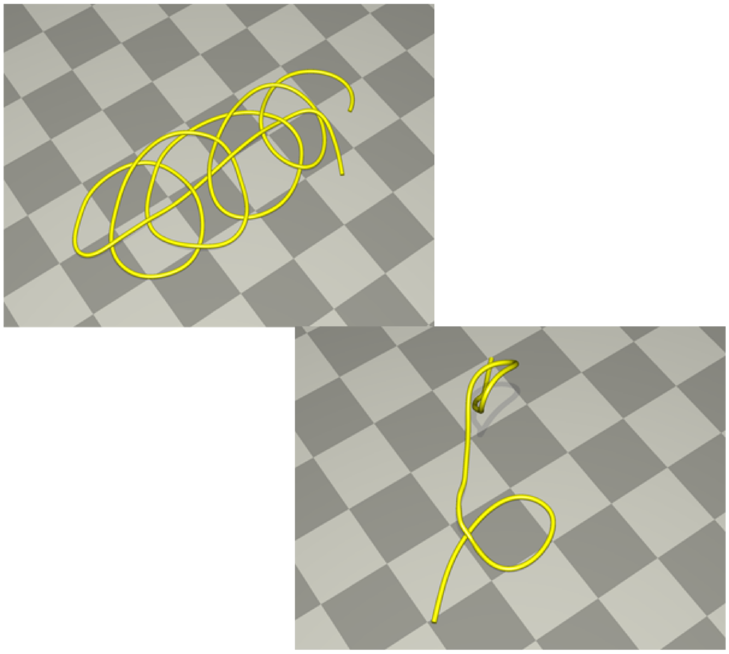
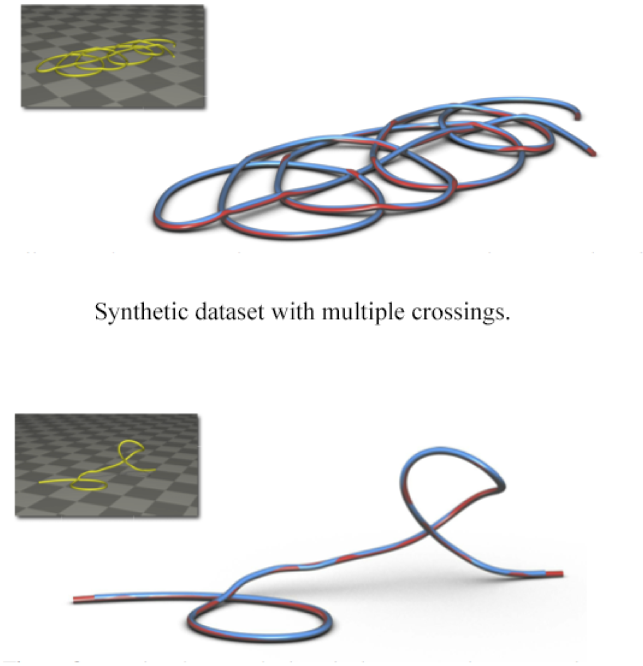


Figure 4.1: Top view of the synthetic datasets.



Synthetic dataset with multiple crossings.

Synthetic dataset with 3D complexity.

Figure 4.2: Comparison between the synthetic datasets (blue) and the output given by the system (red).

4.2 Real datasets (Red Cable 1)

This dataset is the first real challenge to the algorithm. It does not only exhibit complex crossings but also has a nice 3D complexity. The video for this dataset was captured using a professional videocamera with autofocus disabled. 125 images were used to compute the occupancy grid.

The original cable and the result are exposed in figure 4.5. The occupancy grid and initial guesses after the first physical simulation are shown in figure 4.4.

Property	Result
Occupancy Grid	A voxel is considered on if it hits a white pixel in half of the images. The occupancy grid is free of wedges.
Junctions	This cable has complex crossings. The junctions are decently segmented and the system is able to compute nice junction areas without any trouble.
Images	The images are high quality without too much noise and stable illumination between them.
End result	The cable resembles the original.

TABLE 4.2: Result for Red Cable 1.

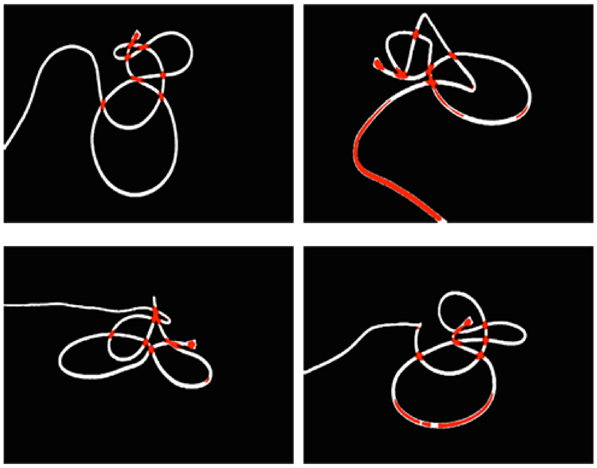


FIGURE 4.3: Segmented images of Red Cable 1.

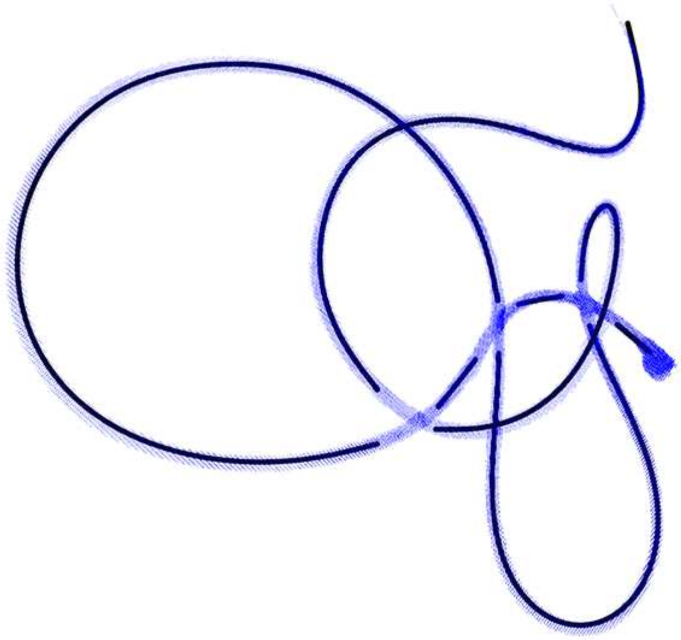


FIGURE 4.4: Occupancy grid and initial guesses before joining the paths.

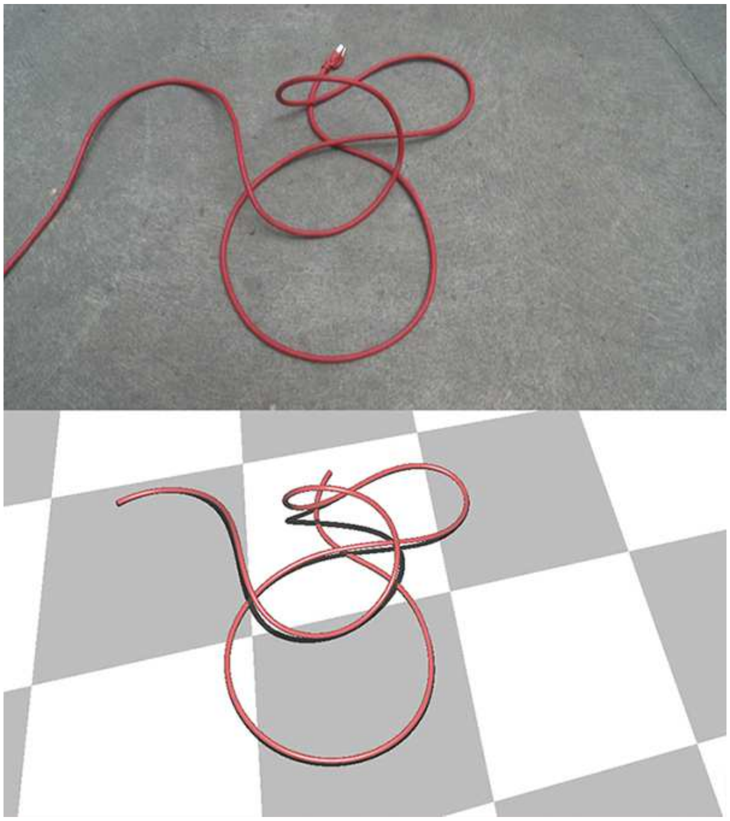


FIGURE 4.5: Comparison between the cable in the video footage (top) and the reconstructed cable (bottom).

4.3 Real datasets (Red Cable 2)

This dataset was captured with the same red cable and same camera as the first dataset. This cable is more complex to process as most parts of the cable are occluded in several images, so it is necessary to be permissive with the constraints when computing the occupancy grid. 95 images were used for this dataset.

Property	Result
Occupancy Grid	A voxel is considered on if it hits a white pixel in a third of the images. The occupancy grid has multiple wedges.
Junctions	This cable has simple crossings, however the quality of the occupancy grid makes it difficult to find the junction areas. Junction areas are found, but they reach a big part of the graph.
Images	The images are high quality without too much noise and stable illumination between them.
End result	The cable presents differences at the junction areas but the cable can be reconstructed with some accuracy.

Table 4.3: Result for Red Cable 2.

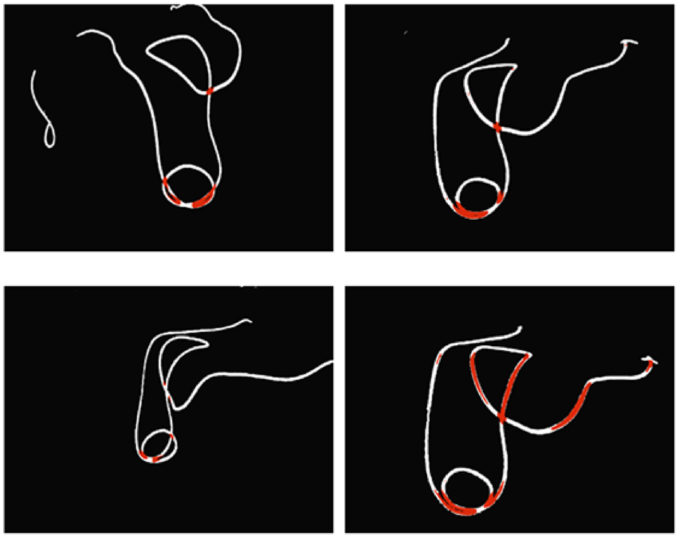


Figure 4.6: Segmented images of Red Cable 2.

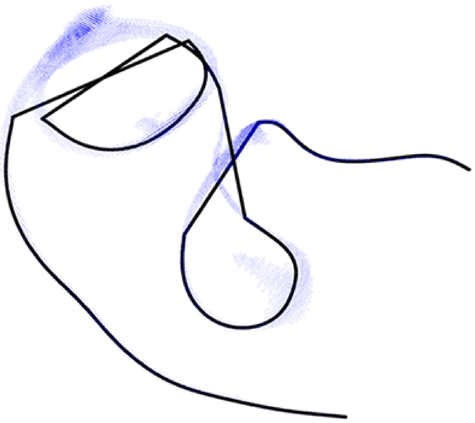


Figure 4.7: Occupancy grid and initial guesses after joining the paths.

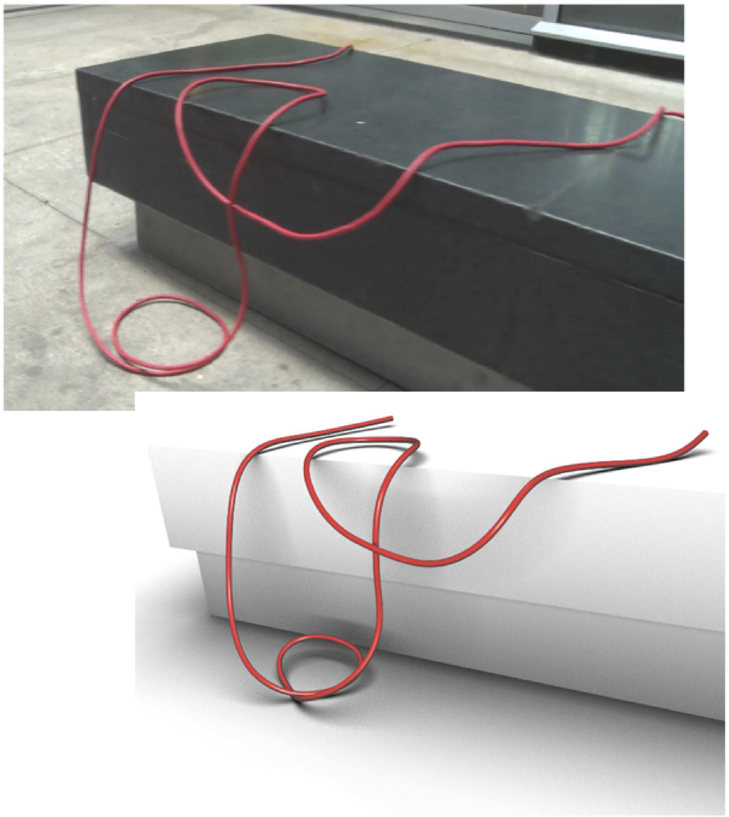


Figure 4.8: Comparison between the cable in the video footage (top) and the reconstructed cable (bottom).

4.4 Real datasets (Blue Cable)

This dataset was captured with a cellphone camera. This cable presents different challenges. The first one, it has many crossings often located very close to each other. The images have a low resolution and they present illumination noise making it difficult for the gradient algorithm to properly function. 100 images were used for this dataset.

A high resolution occupancy grid (400x400x400) was required to find the junction areas. The joining algorithm managed to properly match 16 out of 18 crossings, the other two were corrected using a combinatorial test.

Property	Result
Occupancy Grid	A voxel is considered on if it hits a white pixel in half of the images. The occupancy grid is free of wedges.
Junctions	To properly find the junctions of this cable a high resolution occupancy grid was needed.
Images	The images exhibit noise and are of a lower resolution compared to previous examples.
End result	The cable resembles the original.

TABLE 4.4: Result for Blue Cable.

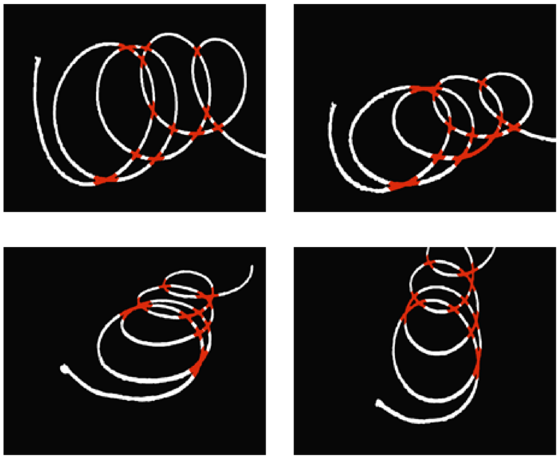


FIGURE 4.9: Segmented images of Blue Cable.

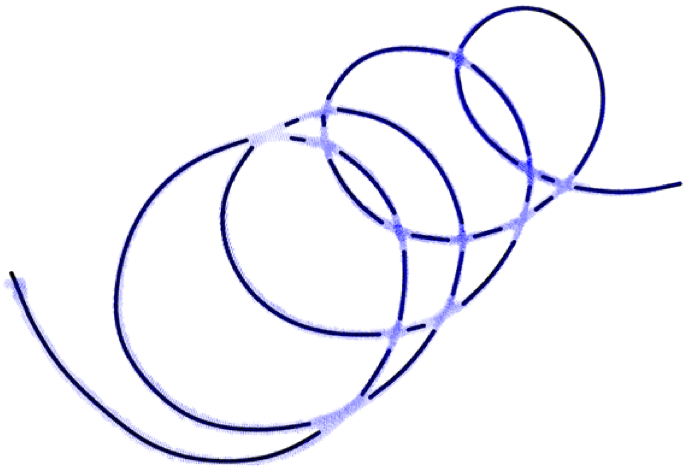


Figure 4.10: Occupancy grid and initial guesses before joining the paths.

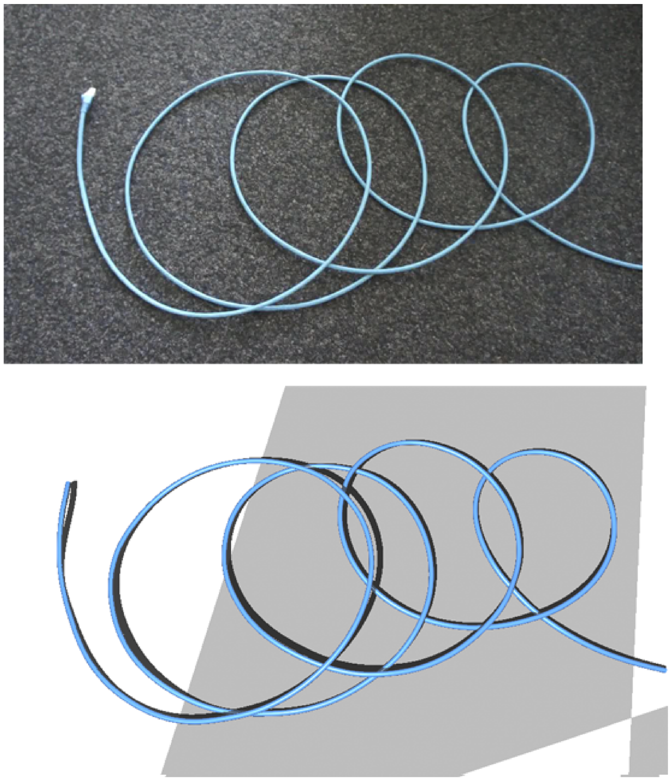


Figure 4.11: Comparison between the cable in the video footage (top) and the reconstructed cable (bottom).

4.5 Evaluation

In this section, the algorithm is evaluated quantitatively in the case of the synthetic datasets and qualitatively for the real datasets.

4.5.1 Quantitative

The synthetic datasets are evaluated quantitatively by comparing the result with the input. Since the cameras used to reconstruct the cable are the same ones used to create the views, both, the input and the reconstructed output should be the same.

Both synthetic cables have a thickness of 1 cm. The result is evaluated by how much the result cable differs from the input cable by averaging the distance from each vertex of the output cable to the closest line segment of the input cable (fig. 4.12).

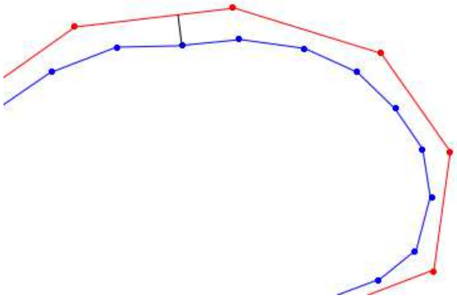


Figure 4.12: Quantitative evaluation. The error of the result (blue) is computed by computing the distance from the vertices to the closest line segment in the input (red). Poly-lines may be sampled differently.

Using this evaluation method, the cable with multiple crossings gave a difference of 3.74 mm which related to its thickness, gives an error of 3.74%. The second cable gave a difference of 1.48 mm which related to its thickness, gives an error of 1.48%. Numerically, the difference is very small.

This error can come from different sources. The biggest contribution comes from the point cloud. The point cloud may contain points that do not belong to the cable and can influence slightly where the centre of the cable is located. Also, during the simulation, the system is set to prefer stiffness of the rod slightly over placing of the rod. The test itself induces a bit of error, since it compares two rods which were built with different sampling.

Qualitative results were given in figure 4.2 of section 4.1.

4.5.2 Qualitative

The real datasets are evaluated qualitatively due to the lack of reference to compare quantitatively. To evaluate these datasets, the resulting cable is put on top of the view and it is measured visually.

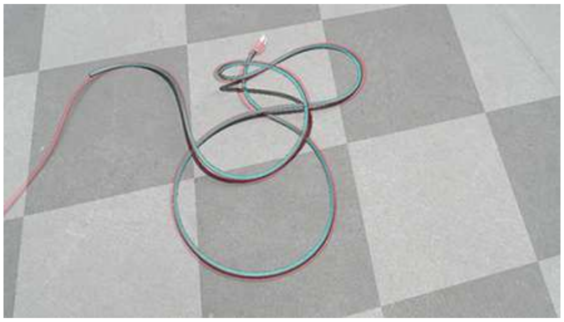


Figure 4.13: The cable resembles topologically the original cable. The red cable is the original cable as exposed in the images, and the blue cable is the reconstructed one.

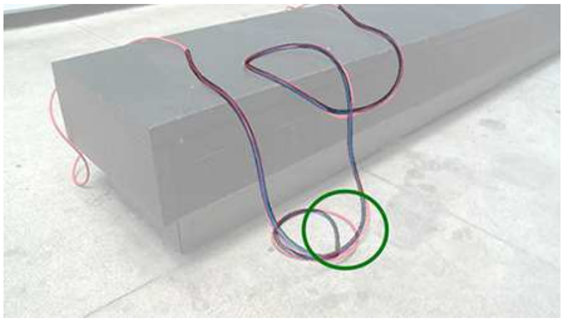


Figure 4.14: The cable resembles topologically the original cable with the exception of one junction area. The red cable is the original cable as exposed in the images, and the blue cable is the reconstructed one.

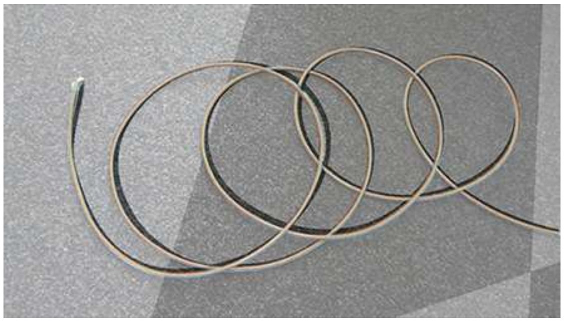


Figure 4.15: The cable resembles topologically the original cable. The blue cable is the original cable as exposed in the images, and the orange cable is the reconstructed one.

4.6 Comparison to L1-medial skeleton

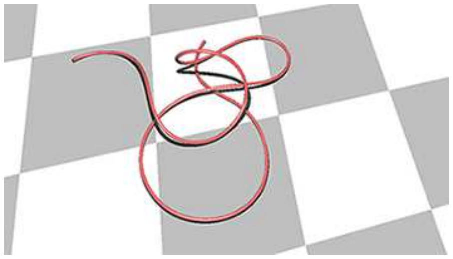


FIGURE 4.16: Red cable result.

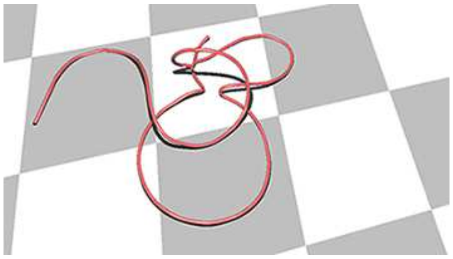


FIGURE 4.17: Red cable L1-medial skeleton (Huang et al. [7]). It can be observed how one junction is completely lost while extracting the L1-medial skeleton of the red cable.

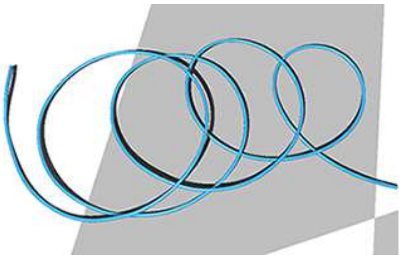


FIGURE 4.18: Blue cable result.

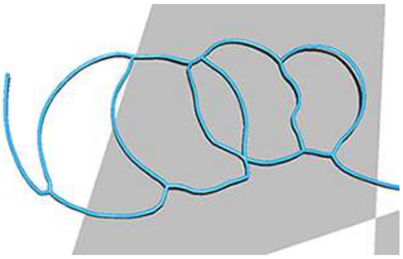


FIGURE 4.19: Blue cable L1-medial skeleton (Huang et al. [7]). In this example, some loops are lost in the skeleton. Also, the junctions are misplaced.

Chapter 5

Conclusion

In this thesis, a method that reconstructs the geometry of tubular structures and establishes their topology from a given set of input images is proposed. First, for each image the 2D topology of the tubular structure is reconstructed identifying the segments and the junctions. This is achieved by a low-level image analysis. Then, the topological information from all the images is fused into a volumetric grid which is the basis for the reconstruction of the 3D structure. The final reconstruction uses physics simulation to connect up the segments. This method is a first step towards robust reconstruction of tubular objects, as those structures are notoriously difficult to reconstruct due to their thin geometry and complex topology. As shown in the results section, this method performs robustly if the segments are clearly visible in at least some views.

5.1 Limitations

As seen in the results chapter, this method performs well on a limited set of junction types. The segmentation can also limit the quality of the end result and it can be further improved. However, the most important limitation is that while many tubular structures such as cables have uniform colour that can be used to solve the topological inconsistencies, certain objects may exhibit a strong texture. A more complex texture is more difficult to segment and also makes the gradient useless as it can not be longer used to connect the paths. These limitations will be addressed in future work.

In detail, the proposed approach has the following limitations:

1. The algorithm fails when two sections of the cable overlap for a long region.

After finding the junction areas, the region between two fronts is treated as belonging to a single cable. However, when the cable overlaps itself for a long region, multiple segments should be considered as part of the same cable. This problem is illustrated in figure 5.1

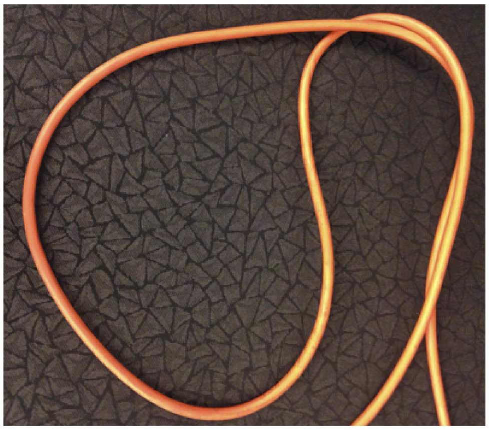


FIGURE 5.1: Overlapping cables. If reconstructed each segment is going to be considered one part of the cable, instead of two, which is the correct assumption.

2. The algorithm can not reconstruct cables with a strong texture. The main problem with strong textures is using the gradient in the final step to connect the initial guesses of the cable. The gradient relies on strong transitions from dark to light and light to dark to assess continuity in overlapping sections of the cable. With a strong texture, strong transitions do not reflect the continuity of the cable anymore, thus a more robust method is needed. An example of a cable with a strong texture is given in figure 5.2.

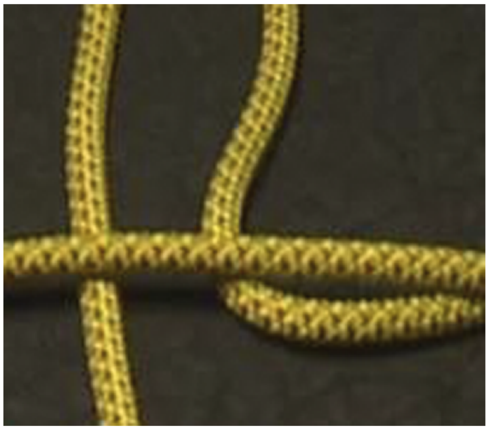


FIGURE 5.2: Cable with a strong texture.

3. Junction areas must be within a certain distance. If the junction areas are too close to each other they will be treated as one and several segments may be lost. There is a workaround for this limitation, which consists in using a denser grid. This limitation is shown in figure 5.3.

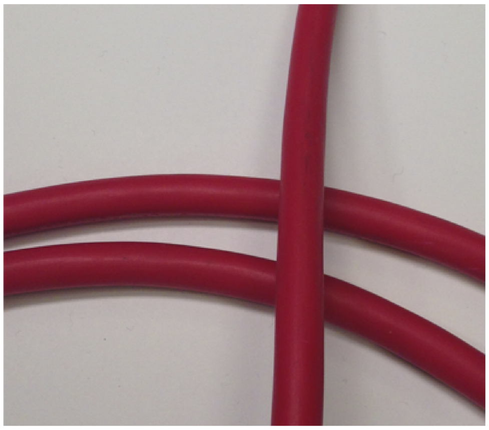


FIGURE 5.3: Junctions close to each other. The algorithm is able to reconstruct this cable, however the grid should be dense.

5.2 Future work

With the limitations exposed in section 5.1, the following solutions are proposed for future work:

1. There are several possible approaches to solve the problem of long overlapping junction areas. One of them is to consider the number of branches per junction area. All the cables reconstructed in this work exhibit junction areas with two or four branches. In The first case can be found if some segment is marked incorrectly as a junction area due to noise in the segmentation. Nonetheless, this case is trivial to solve by just connecting the two end points. The case of four branches is solved using the method exposed in this thesis, connecting first the top part of the cable, followed by the bottom. However, it is possible to have four branches where the cable intersects multiple times. For this case, and other number of branches, it may be possible to find a solution by looking at neighbouring branches.

2. New segmentation techniques can be used to extract a cable with complex texture from the images. One solution is template matching. Since the texture of the cables of this kind is often repetitive, the cable can be extracted by providing the

system with one sample of the texture of the cable and matching it with the whole image. A partial result of this approach is given in figure 5.4.

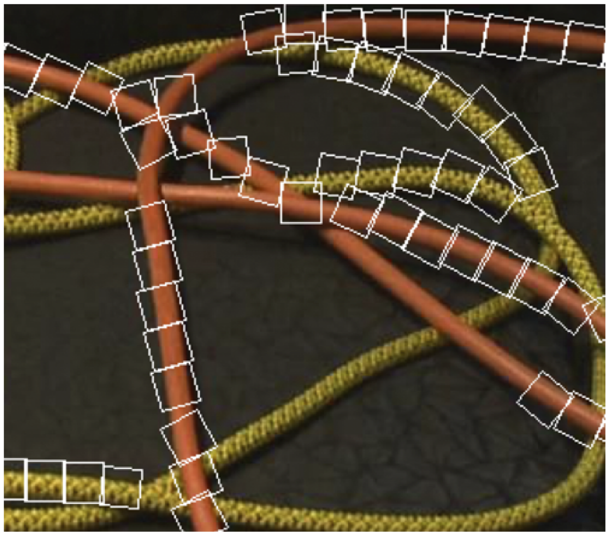


FIGURE 5.4: Segmentation by template matching.

3. The method proposed to segment strong textures can also be used to solve more complex junction areas. If the template matching is done tilting the window at multiple degrees, an orientation of the cable can be obtained such as seen in figure 5.4. It may be possible to use this information to reconstruct the kind of junctions seen in figure 5.5.

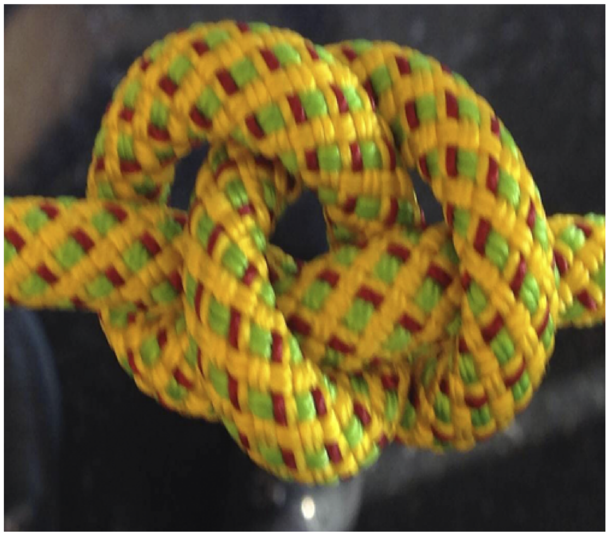


FIGURE 5.5: Knots.

Bibliography

[1] Changchang Wu, J. Frahm, and M. Pollefeys. Repetition-based dense single-view reconstruction. In *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '11, pages 3113–3120, Washington, DC, USA, 2011. IEEE Computer Society. ISBN 978-1-4577-0394-2. doi: 10.1109/CVPR.2011.5995551. URL http://dx.doi.org/10.1109/CVPR.2011.5995551.

[2] Changchang Wu. Towards linear-time incremental structure from motion. In *Proceedings of the 2013 International Conference on 3D Vision*, 3DV '13, pages 127–134, Washington, DC, USA, 2013. IEEE Computer Society. ISBN 978-0-7695-5067-1. doi: 10.1109/3DV.2013.25. URL http://dx.doi.org/10.1109/3DV.2013.25.

[3] Qian-Yi Zhou and Vladlen Koltun. Dense scene reconstruction with points of interest. *ACM Trans. Graph.*, 32(4):112:1–112:8, July 2013. ISSN 0730-0301. doi: 10.1145/2461912.2461919. URL http://doi.acm.org/10.1145/2461912.2461919.

[4] Shahram Izadi, David Kim, Otmar Hilliges, David Molyneaux, Richard Newcombe, Pushmeet Kohli, Jamie Shotton, Steve Hodges, Dustin Freeman, Andrew Davison, and Andrew Fitzgibbon. Kinectfusion: Real-time 3d reconstruction and interaction using a moving depth camera. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, UIST '11, pages 559–568, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0716-1. doi: 10.1145/2047196.2047270. URL http://doi.acm.org/10.1145/2047196.2047270.

[5] Guo Li, Ligang Liu, Hanlin Zheng, and Niloy J. Mitra. Analysis, reconstruction and manipulation using arterial snakes. *ACM Trans. Graph.*, 29(6):152:1–152:10, December 2010. ISSN 0730-0301. doi: 10.1145/1882261.1866178. URL http://doi.acm.org/10.1145/1882261.1866178.

[6] Andrea Tagliasacchi, Hao Zhang, and Daniel Cohen-Or. Curve skeleton extraction from incomplete point cloud. *ACM Trans. Graph.*, 28(3):71:1–71:9, July 2009. ISSN 0730-0301. doi: 10.1145/1531326.1531377. URL http://doi.acm.org/10.1145/1531326.1531377.

[7] Hui Huang, Shihao Wu, Daniel Cohen-Or, Minglun Gong, Hao Zhang, Guiqing Li, and Baoquan Chen. L1-medial skeleton of point cloud. *ACM Trans. Graph.*, 32 (4):65:1–65:8, July 2013. ISSN 0730-0301. doi: 10.1145/2461912.2461913. URL http://doi.acm.org/10.1145/2461912.2461913.

[8] R.J. Valkenburg and A.M. McIvor. Accurate 3d measurement using a structured light system. *Image and Vision Computing*, 16:99–110, 1996.

[9] Brian Curless and Marc Levoy. A volumetric method for building complex models from range images. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '96, pages 303–312, New York, NY, USA, 1996. ACM. ISBN 0-89791-746-4. doi: 10.1145/237170.237269. URL http://doi.acm.org/10.1145/237170.237269.

[10] Liwen Hu, Chongyang Ma, Linjie Luo, and Hao Li. Robust hair capture using simulated examples. *ACM Trans. Graph.*, 33(4):126:1–126:10, July 2014. ISSN 0730-0301. doi: 10.1145/2601097.2601194. URL http://doi.acm.org/10.1145/2601097.2601194.

[11] Yotam Livny, Feilong Yan, Matt Olson, Baoquan Chen, Hao Zhang, and Jihad El-Sana. Automatic reconstruction of tree skeletal structures from point clouds. *ACM Trans. Graph.*, 29(6):151:1–151:8, December 2010. ISSN 0730-0301. doi: 10.1145/1882261.1866177. URL http://doi.acm.org/10.1145/1882261.1866177.

[12] Miklós Bergou, Max Wardetzky, Stephen Robinson, Basile Audoly, and Eitan Grinspun. Discrete elastic rods. *ACM Trans. Graph.*, 27(3):63:1–63:12, August 2008. ISSN 0730-0301. doi: 10.1145/1360612.1360662. URL http://doi.acm.org/10.1145/1360612.1360662.

[13] Heiko Hirschmuller. Stereo processing by semiglobal matching and mutual information. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(2):328–341, February 2008. ISSN 0162-8828. doi: 10.1109/TPAMI.2007.1166. URL http://dx.doi.org/10.1109/TPAMI.2007.1166.

[14] Yasutaka Furukawa, Brian Curless, Steven M Seitz, and Richard Szeliski. Towards internet-scale multi-view stereo. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 1434–1441. IEEE, 2010.

[15] Yasutaka Furukawa and Jean Ponce. Accurate, dense, and robust multiview stereopsis. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32(8):1362–1376, August 2010. ISSN 0162-8828. doi: 10.1109/TPAMI.2009.161. URL http://dx.doi.org/10.1109/TPAMI.2009.161.

[16] Bo Li, Y. V. Venkatesh, Ashraf Kassim, and Yijuan Lu. Improving pmvs algorithm for 3d scene reconstruction from sparse stereo pairs. In *Proceedings of the 14th Pacific-Rim Conference on Advances in Multimedia Information Processing &#151; PCM 2013 - Volume 8294*, pages 221–232, New York, NY, USA, 2013. Springer-Verlag New York, Inc. ISBN 978-3-319-03730-1. doi: 10.1007/978-3-319-03731-8_21. URL http://dx.doi.org/10.1007/978-3-319-03731-8_21.

[17] Yasutaka Furukawa and Jean Ponce. Accurate camera calibration from multi-view stereo and bundle adjustment. *Int. J. Comput. Vision*, 84(3):257–268, September 2009. ISSN 0920-5691. doi: 10.1007/s11263-009-0232-2. URL http://dx.doi.org/10.1007/s11263-009-0232-2.

[18] Maxime Lhuillier and Shuda Yu. Manifold surface reconstruction of an environment from sparse structure-from-motion data. *Comput. Vis. Image Underst.*, 117(11): 1628–1644, November 2013. ISSN 1077-3142. doi: 10.1016/j.cviu.2013.08.002. URL http://dx.doi.org/10.1016/j.cviu.2013.08.002.

[19] Yi Xu and Daniel G. Aliaga. Robust pixel classification for 3d modeling with structured light. In *Proceedings of Graphics Interface 2007*, GI '07, pages 233–240, New York, NY, USA, 2007. ACM. ISBN 978-1-56881-337-0. doi: 10.1145/1268517.1268556. URL http://doi.acm.org/10.1145/1268517.1268556.

[20] Achraf Ben-Hamadou, Charles Soussen, Christian Daul, Walter Blondel, and Didier Wolf. Flexible calibration of structured-light systems projecting point patterns. *Comput. Vis. Image Underst.*, 117(10):1468–1481, October 2013. ISSN 1077-3142. doi: 10.1016/j.cviu.2013.06.002. URL http://dx.doi.org/10.1016/j.cviu.2013.06.002.

[21] Y. Yemez and C. J. Wetherilt. A volumetric fusion technique for surface reconstruction from silhouettes and range data. *Comput. Vis. Image Underst.*, 105 (1):30–41, January 2007. ISSN 1077-3142. doi: 10.1016/j.cviu.2006.07.008. URL http://dx.doi.org/10.1016/j.cviu.2006.07.008.

[22] Peng Song, Xiaojun Wu, and Michael Yu Wang. Volumetric stereo and silhouette fusion for image-based modeling. *Vis. Comput.*, 26(12):1435–1450, December 2010. ISSN 0178-2789. doi: 10.1007/s00371-010-0429-y. URL http://dx.doi.org/10.1007/s00371-010-0429-y.

[23] Marc Alexa, Johannes Behr, Daniel Cohen-Or, Shachar Fleishman, David Levin, and Claudio T. Silva. Point set surfaces. In *Proceedings of the Conference on Visualization '01*, VIS '01, pages 21–28, Washington, DC, USA, 2001. IEEE Computer Society. ISBN 0-7803-7200-X.

[24] In-Kwon Lee. Curve reconstruction from unorganized points. *Comput. Aided Geom. Des.*, 17(2):161–177, February 2000. ISSN 0167-8396. doi: 10.1016/S0167-8396(99)00044-8. URL http://dx.doi.org/10.1016/S0167-8396(99)00044-8.

[25] Benjamin Ummenhofer and Thomas Brox. Point-based 3d reconstruction of thin objects. In *Proceedings of the 2013 IEEE International Conference on Computer Vision*, ICCV '13, pages 969–976, Washington, DC, USA, 2013. IEEE Computer Society. ISBN 978-1-4799-2840-8. doi: 10.1109/ICCV.2013.124. URL http://dx.doi.org/10.1109/ICCV.2013.124.

[26] Chuan-Chu Kuo and Hong-Tzong Yau. A delaunay-based region-growing approach to surface reconstruction from unorganized points. *Comput. Aided Des.*, 37(8):825–835, July 2005. ISSN 0010-4485. doi: 10.1016/j.cad.2004.09.011. URL http://dx.doi.org/10.1016/j.cad.2004.09.011.

[27] P. Alliez, D. Cohen-Steiner, Y. Tong, and M. Desbrun. Voronoi-based variational reconstruction of unoriented point sets. In *Proceedings of the Fifth Eurographics Symposium on Geometry Processing*, SGP '07, pages 39–48, Aire-la-Ville, Switzerland, Switzerland, 2007. Eurographics Association. ISBN 978-3-905673-46-3.

[28] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *Proceedings of the Fourth Eurographics Symposium on Geometry Processing*, SGP '06, pages 61–70, Aire-la-Ville, Switzerland, Switzerland, 2006. Eurographics Association. ISBN 3-905673-36-3.

[29] Kun Zhou, Jin Huang, John Snyder, Xinguo Liu, Hujun Bao, Baining Guo, and Heung-Yeung Shum. Large mesh deformation using the volumetric graph laplacian. *ACM Trans. Graph.*, 24(3):496–503, July 2005. ISSN 0730-0301. doi: 10.1145/1073204.1073219. URL http://doi.acm.org/10.1145/1073204.1073219.

[30] Gregory M. Nielson. Dual marching cubes. In *Proceedings of the Conference on Visualization '04*, VIS '04, pages 489–496, Washington, DC, USA, 2004. IEEE Computer Society. ISBN 0-7803-8788-0. doi: 10.1109/VISUAL.2004.28. URL http://dx.doi.org/10.1109/VISUAL.2004.28.