University of Cape Town

Using Multiple View Geometry for Transmission Tower Reconstruction

Author:
Bhavani Morarjee

Supervisors:
Dr. F.C. Nicolls and Prof. E. Boje

A dissertation submitted in fulfilment of the requirements for the degree of Master of Science in Electrical Engineering

in the

Department of Electrical Engineering

May 2016

Declaration of Authorship

I, Bhavani Morarjee, declare that this dissertation titled, 'Using Multiple View Geometry for Transmission Tower Reconstruction' and the work presented in it are my own except where otherwise stated. I confirm that:

- This submission is towards a degree of Master of Science in Engineering at the University of Cape Town.
- This dissertation has not been submitted before for any degree at any other university.
- I know the meaning of plagiarism and declare that all the work in the document, save for that which is properly acknowledged, is my own.

Signed: ________________________________

Date: ________________________________

"The only thing worse than being blind is having sight but no vision."

Helen Keller

UNIVERSITY OF CAPE TOWN

Abstract

Department of Electrical Engineering

Master of Science in Electrical Engineering

Using Multiple View Geometry for Transmission Tower Reconstruction

by Bhavani Morarjee

Automated platforms that conduct power line inspections need to have a vision system which is robust for their harsh working environment. State-of-the-art work in this field focuses on detecting primitive shapes in 2D images in order to isolate power line hardware. Recent trends are starting to explore 3D vision for autonomous platforms, both for navigation and inspection. However, expensive options in the form of specialised hardware is being researched. A cost effective approach would begin with multiple view geometry. Therefore, this study aims to provide a 3D context in the form of a reconstructed transmission pylon that arises from image data. To this end, structure from motion techniques are used to understand multiple view geometry and extract camera extrinsics. Thereafter, a state-of-art line reconstruction algorithm is applied to produce a tower. The pipeline designed is capable of reconstructing a tower up to scale, provided that a known measurement of the scene is provided. Both 2D and 3D hypotheses are formed and scored using edge detection methods before being clustered into a final model. The process of matching 2D lines is based on an exploitation of epipolar geometry, where such 2D lines are detected via the Line Segment Detection (LSD) algorithm. The transmission tower reconstructions contrast their point cloud counterparts, in that no specialised tools or software is required. Instead, this work exploits the wiry nature of the tower and uses camera geometry to evaluate algorithms that are suitable for offline tower reconstruction.

Acknowledgements

I humbly thank my supervisors Prof. E. Boje and Dr. F.C. Nicolls for their academic advice and insight provided throughout this project. I am also very grateful to Mayuresh Kulkarni, Yashren Reddi and Rick Bosch for their technical help and moral support.

Work by Manuel Hofer and his team at TU Graz has formed the foundation of this research and their data was graciously provided to consolidate the concepts discussed in this study.

Contents

List of Figures

List of Tables

Symbols

C	Camera centre	3×1
$CN1, CN2$	Cylinder cap normals	3×1
CoG	Centre of gravity (3D)	3×1
$CP1, CP2$	Cylinder cap ends	3×1
d_{ang}	Neighbourhood threshold	deg
d_c	Neighbourhood threshold	metres or mm
$\vec{dir}$	Line segment orientation	3×1
E	Essential Matrix	3×3
H_{min}	Clustering threshold	
K	Camera calibration matrix	3×3
$\mathbf{l}, \mathbf{l}'$	Epipolar lines	(homogeneous) 3×1
l_i	2D line	3×1
L_m, L_n	3D Line segment	3×1
$\mathbf{P}$	Pinhole camera matrix	3×4
$\mathbf{R}$	Rotation matrix	3×3
r	Clustering radius	mm (or cms)
S	Set of 3D hypotheses	
s	Line segment score	
$\mathbf{t}$	Translation	3×1
$\mathbf{X}$	Homogeneous point in 3D	4×1
$q, \mathbf{x}, \mathbf{x}'$	Image points	(homogeneous 3×1)

For Nivedita . . .

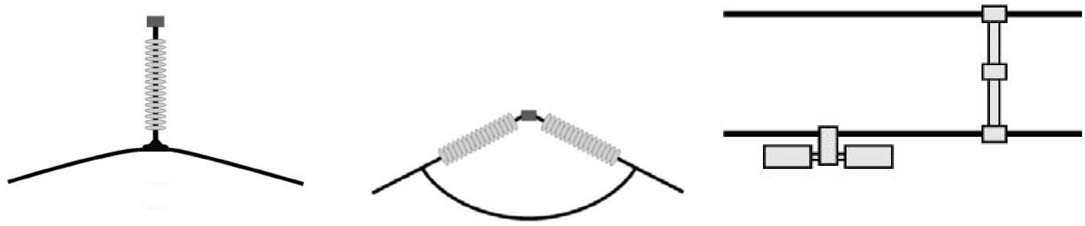
Chapter 1

Introduction

1.1 Research motivation

Image processing can be integrated into a multitude of scenarios, one of which is in the growing power sector in South Africa. To elaborate, there is an increase in energy demand, and this leads to the dedication of more resources towards power line and equipment inspection/maintenance. However, South Africa has close to 30 000 km of transmission lines [5] and the large, financial expense of inspecting the integrity of transmission line components is coupled with the danger that this task poses to skilled personnel. Current inspections are performed manually by flying helicopters and fixed-wing aircraft over the lines, or by patrolling with ground personnel. These tasks are necessary, especially since some of the components have been in commission for several decades. However, this process can be streamlined if skilled employees were directed to problematic power line equipment only. Less time should be dedicated towards the tedious search for hazardous components from thousands of inspection images. This is why power line inspection via UAVs and robots is studied. Their potential to improve inspection reliability, and reduce both cost and danger, is attractive.

However the scope of a power line inspection robot should include navigation along a live transmission line. Thereafter, inspection of different hardware can follow. Two such robots that are striving to achieve these tasks are UKZN's Power Line Inspection Robot [27] and Hydro-Quebec's LineScout [46]. Recognising specific hardware components can initiate appropriate movement sequences and can make inspection routines more autonomous. Currently, these platforms are prototypes that require human intervention to ensure that the robot does not fall from the line. The interesting challenge lies in developing computer vision strategies to increase autonomy. Visual sensing may provide a way for the robot to track targets and inspect components. When reconstructing 3D models, image processing methods should be chosen on the basis of their ability to exploit the physical construct of target components.



(A) Suspension insulator. (B) Strain insulators. (C) Damper and spacer.

FIGURE 1.1: Important power line components [21].

1.2 Problem identification

Most robotic platforms jump straight to employing specialised algorithms (based on images) to detect the presence, and possible damage, of specific components. Although these activities are vital, they have overlooked the development of robot autonomy as far as incorporating 3D models is concerned. These 3D models can be used to localize a robot before realizing specific line components as identified in Figure 1.1. In addition, GPS can be wrong up to a few metres while onboard sensors are sensitive to drift. The idea of having an inspection robot use 3D models should not be overlooked given that their working environment is a structured, man-made context.

A feasible example of a 3D target is a model of a transmission pylon because of its size, repeatability (intermittent locations) and rigid construct. A transmission line tower is a 'wiry' structure and thus rich in geometric content, making it a suitable candidate for generating a 3D model. It is considered wiry because of its beams configured in an organised construct. The tower is significantly tall (for example, a 220kV tower can be 30 m tall) and if it can be tracked relative to the robot's position, the robot can subsequently zoom in on specific line equipment to perform inspections. By using camera models and projective geometry, the robot would only consider the pixels it needs in order to locate power components such as those in Figure 1.1.

Most 3D representations for this possible target object are in the form of CAD drawings and point clouds, both of which require specialised software and tools to generate. These consume a significant portion of disk space. This leads to the question of what makes a good tower model. Point cloud data from laser scanners require the use of expensive equipment. The resulting model still needs to be processed in specialised software [18] and it can take hours to render a model like the one depicted in Figure 1.2 [43]. The sizes of point cloud files are also in the hundreds of megabytes.

A CAD model may also seem to be a likely candidate. However, the robot will be working outdoors with varying lighting conditions, occlusion and other unwanted objects in the scene. The effect of synthetic textures in a CAD tower model may not be ideal for when the robot

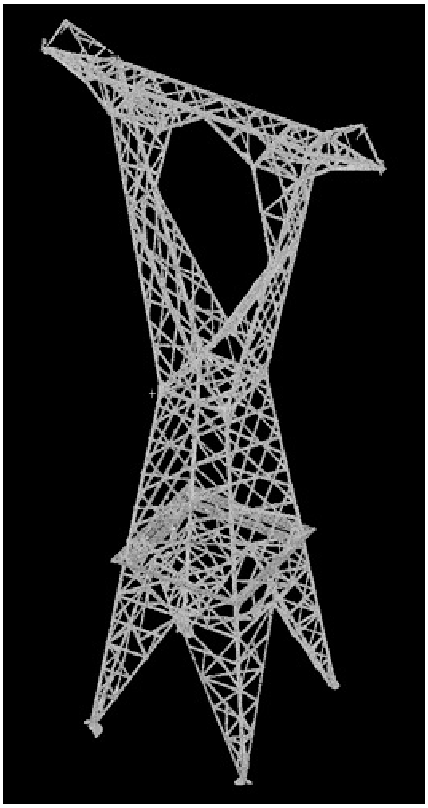


FIGURE 1.2: Example of a point cloud after rendering [43].

needs to track a real tower (i.e. comparing the surface finish of tower beams between CAD models and acquired images).

It is also possible that current models are just records on paper, capturing only a general sense of tower geometry. Some towers have asymmetrical cross-arms due to lines changing directions or asymmetrical legs due to an uneven terrain. Towers have to conform to their local landscape and will also vary in height depending on application. Tower heights, for 400kV transmission lines, can range between 28 m and 60 m [39].

There is no clear indication of how a 3D transmission tower can best be reconstructed from images. Available algorithms tackle subsets of these problems in the form of edge detection and line detection, for instance. Feature tracking is limited to popular algorithms that find interest points that are not related to the target object itself. Robotic platforms in the power industry start with a target object that is desired and contrive algorithms to build an application, as opposed to consulting existing multiple view geometry techniques and questioning how these can best suit a reconstruction problem. This work is an effort to answer the question,"What combination of image processing tools and state-of-art algorithms exist to facilitate a tower reconstruction pipeline?"

1.3 Research objective

This work is more concerned with image processing rather than specific robotic platforms that work on power lines. Therefore, any algorithm studied and implemented must have an outcome that can benefit a UAV, robot, helicopter or ground crew. This research aims to reconstruct a 3D tower model from multiple 2D images and this requires understanding projective geometry as well as building and using camera models. Feature points should rely on the geometry of the desired object and the process can be an offline procedure. Comparisons between the reconstruction and its 3D counterpart will be made so as to determine the strengths and weaknesses of the algorithms chosen.

This research will:

- Describe how state-of-art applications use image processing tools for power line inspection.
- Based on state-of-art review, show how a reconstructed tower may be obtained from understanding and exploiting camera geometry offline.
- Finally, compare a tower reconstruction to its physical counterpart, thereby exposing any challenges that are faced in the reconstruction process.

The literature review in Chapter 2 identifies the visual approaches used by existing platforms, acknowledges state-of-art algorithms for detecting power line hardware and introduces algorithms that concern feature detection and structure from motion. The pinhole camera model and epipolar geometry are concepts of projective geometry explained in Chapter 3. This knowledge is used in the chosen reconstruction process, outlined in Chapter 4. Chapter 5 describes the outcome of applying the selected reconstruction strategy. Finally, concluding remarks and future work that extends the ideas presented in this research are provided in Chapter 6.

Chapter 2

Literature review

To understand how image processing can facilitate power line inspection, state-of-art inspection prototypes must be acknowledged. This literature review discusses the computer vision strategies adopted by two such robots, the PLIR and LineScout. Next, a discussion about algorithms that have been designed to detect power line components follows. This is to address how current methods are based on shape primitives (such as circles, lines and ellipses) and edge detection in order to distinguish between obstacles. Finally, the structure from motion problem is mentioned to introduce how multiple view geometry is concerned with projective geometry and 3D reconstructions.

2.1 UKZN's Power Line Inspection Robot (PLIR)

An exemplary robot in the field of autonomous power line inspection was designed by [27] at UKZN. This platform may be equipped with a multi-spectral camera [44] that allows the robot to detect corona from partial discharge in the UV spectrum, as well as faults from infra-red (and visual) data. Information from images was investigated in the early stages of PLIR's development [27]. It was acknowledged that cameras would aid the manoeuvrability of the robot when bypassing obstacles along a power line [27]. The need to identify primitive features in images – to suggest the type of component present – was based solely on pixel information. However, this introduced complications about how to describe pixels that possibly indicated components.

In [27], it was mentioned that reconstruction from 2D images could ensue if camera calibration and point correspondences were studied. The image processing algorithms investigated did not strive towards building reusable models. The task of identifying components was a straightforward application of algorithms on the images themselves. An underlying principle behind the image processing strategies, conducted in [27], was exploiting pixel intensities. Furthermore, the hierarchy described in [27] was based on an important assumption that a conductor line would always be visible in the camera's field of view.

Visual processing in [27] used a Sobel edge [29] detection mechanism. This decision was based on the smoothing aspect of the Sobel operator and its ability to attenuate noise. Morphological filters [13] were used if the edges appeared broken. This method of inferring a line was ambiguous and deprived the robot from working with a repeatable, and therefore reliable, perception of an object. Occlusions and unexpected obstacles were not catered for. Nevertheless, using these now-completed edges, the Hough transform was subsequently applied [12] to parametrise edges and circles from the detected edges. The specific need to detect circles in [27] was to identify damper ends on a line. The reliability of detecting these dampers depended on the robustness of the edge detection, and broken lines (or edges) were often detected despite morphological filtering. The straightforward eroding and dilating of pixels was not related to structural features of components, but rather pixel information. For a power line inspection robot, a more definitive decision about an obstacle's presence is necessary. The author of [27], Lorimer, constructed an image processing pipeline that relied on an ordered arrangement of 2D operators and filters, all of which could be affected by lighting or occlusion.

However, [27] did attempt to reconstruct feature points by using a stereo camera pair to ascertain the depth of the scene. This was to acquire 3D information based on a particular algorithm's set of feature pixels. It was mentioned in [27] that the Hough transform applied was insufficient and better reconstructions were desired. After recent correspondence with Lorimer [26], it became known that the robot, at present, does not incorporate the above-mentioned techniques during its demonstrations on an actual transmission line, outdoors and under varying lighting conditions. The nature of the features detected in images would vary and robot control is critical. Since the work of [27], improvements towards the PLIR are continously being studied. Figures 2.1A and 2.1B show the PLIR.

(A) PLIR robot.

(B) PLIR on a transmission line test in New Zealand.

FIGURE 2.1: South Africa's PLIR robot [26].

2.2 Hydro-Québec robots and developments towards 3D vision

Hydro-Québec developed three different robots since 1998 for transmission line maintenance. LineROVer was the first robot introduced by the group and was deployed on a real grid in 2000. It was a tele-operated system that did not perform autonomous object recognition or classification. It simply had visual cameras as well as an IR camera to detect overheating components [32]. The second robot was designed to bypass obstacles but no major contribution to computer vision or object classification was presented. The most recent robot from Hydro-Québec, the LineScout, debuted in 2006 and is a tele-operated platform designed for future autonomy. LineScout is made aware of target locations along the transmission line and thus points its camera in appropriate directions [46].

FIGURE 2.2: The LineScout robot on a transmission line [46].

FIGURE 2.3: An older robot by Hydro-Québec, the LineROVer, with a mounted UTM-30LX laser range finder [41].

2.2.1 LineScout and the UTM-30LX LIDAR system for obstacle detection

A laser range finder is incorporated into the present LineScout's design. First, a justification for using a LIDAR (Light Detection and Ranging) mechanism was given in [38], suggesting that the device is unmatched in terms of data resolution, price, long detection ranges (up to 30 m) and acquisition speed. The work of [38] investigated how well the LIDAR system could operate with changing distance, surface finish and sensor orientation, and from mock-experiments distances below 500 mm were under-estimated while distances above 1500 mm were over-estimated. It was reported that sensor orientation did not infringe on data acquisition when observing non-cylindrical targets. In the case of cylindrical targets, the incident laser beam started to affect the distance error significantly i.e. if conductors were viewed at angles further away from 90 degrees. It was then concluded in [38] that despite these variations in results, parameters could be set (appropriate distances and angles) to allow the UTM-30LX LIDAR system to be present onboard the power line inspection robot.

The follow-up to work presented in [38] involved the setting of thresholds to detect specific obstacles under outdoor-like conditions such as high temperatures and vibrations [41]. The parameters were apparent distance, perceived diameter and signal intensity. Temperature and lighting were factors that influenced these measurements but coupled with a false-positive rejection mechanism, [41] was confident that splices, dampers and clamps could be detected using the LIDAR system.

In Figure 2.3, the LIDAR system is seen to be small and compact enough to be mounted on a robot. The cost of one UTM-30LX system is $5000 [17].

2.3 Power line robots relying on visual data for navigation

The Chinese Academy of Science (CAS) developed a two-fold control strategy for a power line inspection robot [50]. An autonomous mode would be used for navigation when a particular obstacle is recognised in order to initiate appropriate movement sequences, while direct control (tele-operation) is necessary if the robot encounters a new type of obstacle. These strategies make use of a video camera and all scenarios work with a robot positioned on an overhead ground wire. Use of databases contributes to autonomous manoeuvrability in [50] as stored targets are referenced by order of towers.

This method of incorporating intelligence in an inspection robot is more robust than [37] who suggested that a line crawling robot should have multiple proximity sensors. These sensors would have pre-defined thresholds for different classes obstacles. The control strategy presented by [50] is more enriching to the robot, especially since camera feeds serve as better records than simply a set of thresholds.

More research into obstacle classification, particularly for robotic power line inspection, was conducted by the CAS [8, 9]. Their robot is equipped with Charge Coupled Devices (CCDs) and computer vision efforts are focused on detecting 2D features (lines, ellipses and circles) to discriminate between components such as insulator plates (see section 2.4.1).

2.4 Detection of primitive shapes to infer power line components

State-of-art work, such as [8] and [23], is mainly based on primitive shape detection. Primitive shapes are detected in images and can infer the presence of certain power line components. In this section, work is presented to expand on how these shapes are detected and how components are suggested based on these findings. The quality of these detections is discussed. Thereafter, a line-detection scheme [10] is discussed as it is a popular 2D line detection algorithm.

2.4.1 Damper and insulator detections

The work of [8] was based on devising an algorithm to classify three types of obstacles. These were dampers, and suspension and strain insulators. Extracting lines in 2D images forms the backbone of this algorithm since circles, arcs and ellipses are used to describe targets. Insulator strings appear as circular objects on the camera focal plane, and the suspension clamp appears as an ellipse. Figure 2.4 illustrates the detection of an insulator plate. The authors claimed the features to be unique and therefore manipulated the problem statement into finding circles

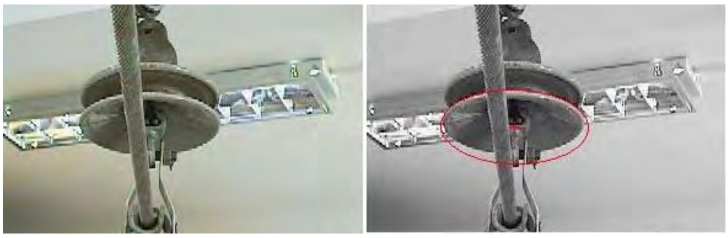


FIGURE 2.4: Detection of a plate on a suspension insulator [8].

and ellipses. More qualitative performance measures are required, however, especially under varying situations such as change in illuminance and scale. It was not mentioned in [8] how background clutter and occlusion would affect results. Figure 2.4 reveals how a plate on a suspension insulator is detected. This position is ideal for detecting a circle and this algorithm is therefore view-dependent. Image features of power line towers (and poles) were investigated for real time tracking [45, 49]. Understanding these methods reveals how corner points and simple shapes (for example, lines) are detected to infer the presence of power line components. The Scale Invariant Feature Transform (SIFT) algorithm [28] is also discussed even though it wasn't designed specifically to track transmission towers.

2.4.2 Conductor and spacer detection

The work of [23] involves automatic image processing for conductors and spacers. The system was designed in OpenCV [34] and is actually a sequential set of algorithms. First, conductor localisation is performed before the spacer detection module can be accessed. The system relies on being given an expected number of conductors to be localised. Any image containing a conductor is passed on to the spacer-detection module. The purpose of this architecture is not to negate manual inspection of the spacer component, but to reduce the number of images to be processed by skilled workers. Besides the specification of the number of anticipated conductors, no other prerequisites are required for inspection.

Conductor Localisation: The conductor localisation module in [23] has the task of extracting lines from digital images. It was decided that template matching would be the strategy used for its capability to ignore background features, which are classified as background noise. It was also reasoned that popular alternatives such as the Hough Transform would have detected too many lines, and thus nullified the objective of allowing the system to be an automated inspection tool.

Spacer Detection: Following the successful identification of conductors, the spacer detection algorithm returns a bounding rectangle to signify the presence of a spacer on a quad-conductor. To achieve this, the image from the previous step is cropped and rotated, and is subjected

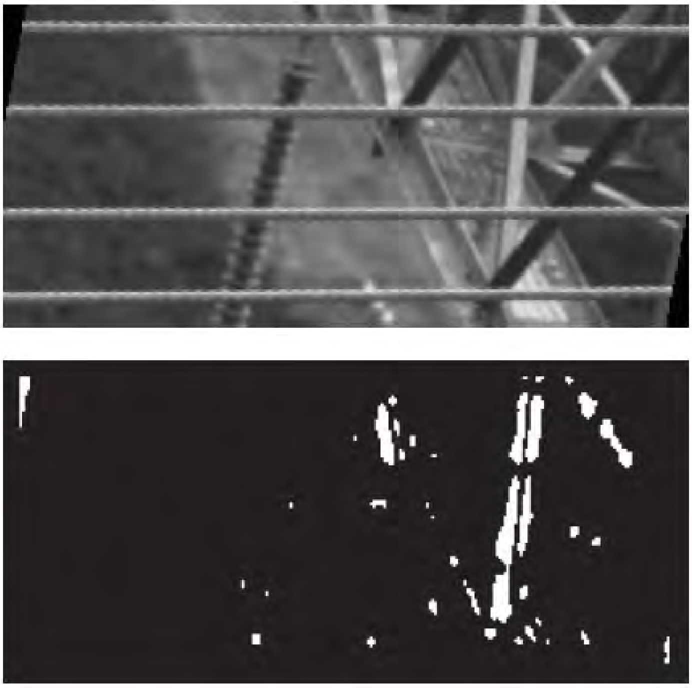


FIGURE 2.5: False recognition of a spacer [23].

to a Gabor filter [31] for feature determination. Results from the filter, referred to as the spacer-Gabor images, indicate if spacers are present (where a large cluster of pixels are found).

False positives for the spacer detection module were reasoned to be caused by intrusion of dampers, which after Gabor filtering may appear as spacers. Another cause was a conductor changing/terminating in mid-frame. Missed detections were caused by the conductors and spacers being too far away from the camera, and partial capture of components in some of the processed images. The possibility of background features infringing on the accuracy of the spacer detection module is not avoidable and is still high despite the justification for using template matching (see Figure 2.5). The spacer detection system depends on the conductor localisation algorithm. There are many variables to consider in order to get a robust pipeline working in this system.

In Figure 2.5, part of the tower is viewed in a way that the algorithm incorrectly classifies the scene as containing a spacer.

2.4.3 Distribution pole detection

The experiments in [49] employed image processing techniques on distribution pylons (or poles) and not transmission towers. Surveillance was performed from a camera mounted on a helicopter. The objective was to quickly establish the presence of a pole in the FOV (field of view) and subsequently locate it in the image. The pole was to be tracked in real time as the helicopter moved. Feature extraction was necessary to isolate the object of interest in an image, especially from background clutter. This called for emphasising the pole's features and suppressing those of the background to determine the coordinates of the pole-top in the image plane. Since the pole structure is known *a priori*, this was exploited in the recognition process as the pre-emptive knowledge acted as a model to which image features were compared. These 2D features were actual corner points that belonged to the pole.

The tracking algorithm in [49] produced estimated coordinates of the target and demonstrated the feasibility of using video surveillance. However, success rates were reported to range from 65-92% on real images. This success rate is too erratic and can lose the target in the event of zoom, while there is an additional limitation based on viewpoint [11]. Despite these shortcomings, the knowledge gained from [49] is that using corner points (structural, model-based features) is a feasible alternative to feature-based key points.

2.4.4 Automatic detection of electricity pylons in aerial video sequences

By using a 2D IIR (Infinite Impulse Response) filter [51] as well as the Hough transform [1], the work in [45] involved constructing a line-detector for locating power pylons. The IIR filter processed data in the x and y directions, as well a z (diagonal) direction. The corresponding filters, for each direction, were $X(x, y)$, $Y(x, y)$ and $Z(x, y)$. By convolving the filter with image I, lines of interest were found. The next stage involved employing the Hough transform for straight line parametrization before dividing the image into regions. The number of lines found in each region was captured. This was done to obtain a confidence value that related a particular region to the presence of a power pylon. The work of [45] capitalised on the fact that the structure of the pylon consisted of multiple lines. The pylon detection scheme in [45] achieved a 97% true positive rate and a false positive rate of 40%. However, if the false positive rate was reduced to 18%, the true positive rate was 60%. The downfall of this algorithm is its inability to filter out unwanted lines. It could also not account for occlusion or ambiguity (roads were detected as pylons).

2.5 Algorithms for feature detection

Feature detection is work primarily concerned with resources in the image space. Points of interests that may be viewed repeatedly need to be described and tracked robustly. Detecting lines in a 2D image is a well known edge detection problem and a solution is often required based on the context of the application. Acknowledging the algorithms that achieve these goals is fundamental to building reconstruction pipelines. The most suitable algorithms for detecting edges and other feature points can influence the outcome of a 3D reconstruction. The fact that primitive shapes about a rigid, man-made object can be detected (as described in the previous section) calls for these algorithms to be scrutinised further.

2.5.1 SIFT

A feature or keypoint is defined as an interest point that can be found repeatedly under varying conditions. A popular feature detector is the Scale Invariant Feature Transform (SIFT) algorithm [28]. Its parameters are fixed to work on all images [28]. The SIFT algorithm is invariant to scale and partial illumination, and is also robust to rotation. The algorithm described in [28] consists of identifying extrema across scaled images and localizing keypoints. These keypoints serve as interest points and are described via keypoint descriptors. The descriptors are necessary for matching keypoints across multiple images. However, the algorithm by itself is a blanket solution to finding common points in several images and false positives are still prevalent. For images of wiry targets (towers) that lack texture, the SIFT algorithm may propose false correspondences. Another important observation to acknowledge is that descriptors for feature points are not strictly limited to the geometry of the tower itself. Background detail with texture are not prevented from being feature points. Therefore, relying on this algorithm alone would not be ideal when a specific target (tower) is to be tracked outdoors. This popular algorithm is available in libraries such as OpenCV [34]. The outcome of using SIFT on two images of a model tower is shown in Figure 2.6.

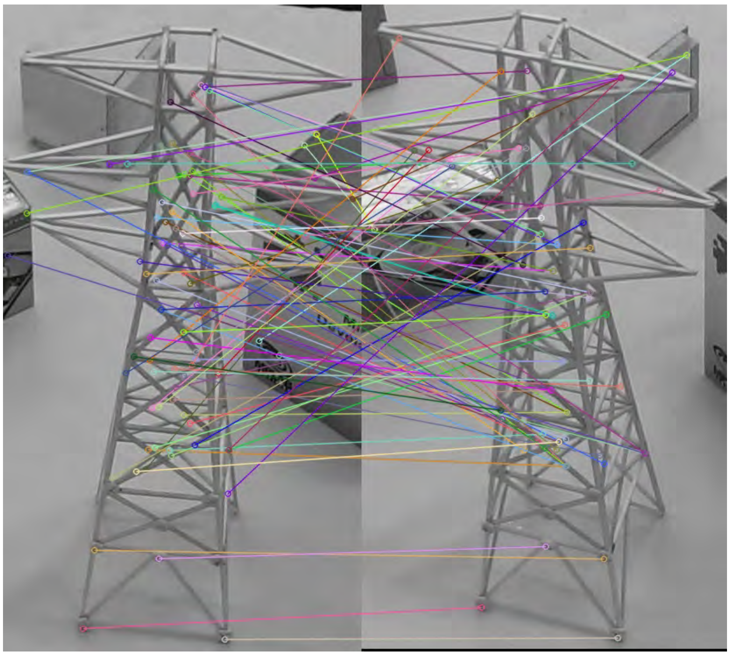


FIGURE 2.6: Example of matches between images using SIFT from the OpenCV library [34].

There are false matches in Figure 2.6 which would not be ideal for a robotic platform, especially if a level of autonomy is to be achieved.

2.5.2 Edge detection with Hough lines to detect primitive shapes

The work in [52] is similar to [8] where features of obstacles are used to recognise power line components. Primitive shapes such as lines, circles and ellipses are used to identify hardware components such as insulators and clamps. A hierarchy is set up as part of the robot's control mechanism together with a ground PC, making remote image processing possible. Each side of the robot had 2 cameras, and it is expected that the bottom ends of insulator strings would appear as ellipses while circles could be indicative of dampers and clamps. The component-recognition process thus resorts to looking for circles and ellipses in captured images, to identify 3 types of obstacles i.e. counterweights, suspension clamps and strain clamps.

In [52], the general construct of power lines is used to influence the operation of the algorithm. For example, in an image, if a straight line runs from the top to the bottom, this was indicative of a power line. Insulators assist the recognition process because of their attachments to the clamps and line. The shapes themselves are identified by use of Sobel and Canny edge detectors as well as the Hough transform.

However, [52] does not mention the possibilities of ambiguity in their strategy, how dependent the shape identification is on camera orientation, and how occlusion affects the end result. These scenarios are often encountered by a power line inspection robot, together with background clutter and poor lighting.

2.5.3 Line segment detector (LSD)

By analysing the orientations of pixel intensities in a small image region, the algorithm in [10] can detect locally straight segments quickly and with no intervention by the user (parameter-tuning). The algorithm begins by describing certain areas or zones, in an image, as the areas with a sharp contrast in intensity levels. The direction of these transitions result in gradient lines and corresponding level lines. Note that in Figure 2.7, the level line is perpendicular to the gradient. Acquiring the level line angle for each pixel then contributes a level line field (which is a unit vector field). Given a certain tolerance τ on the level line angle, connected regions in the field may be constructed by grouping pixels that have almost equal level line angles. In [10], these regions are known as line support regions, where each region is a set of pixels that could potentially be a line segment. Figure 2.9 illustrates the idea behind a line support region. In Figure 2.9, the rectangle contains only 8 pixels that have level lines oriented in the same direction as that of the rectangle. To validate the support region the Helmholtz principle [3] is used, suggesting that a noisy image should not produce detections.
LSD is a highly-recommended algorithm because it requires no external tuning of parameters, is fast, robust and works on all images [24]. The authors of the LSD algorithm [10] mention that

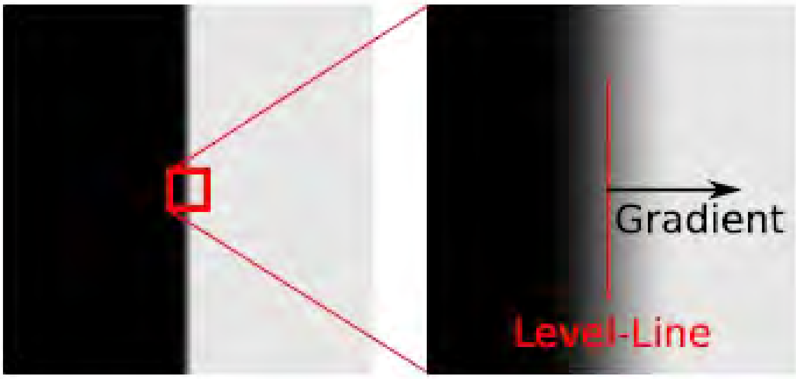


FIGURE 2.7: Example of an image gradient and corresponding level line [10].

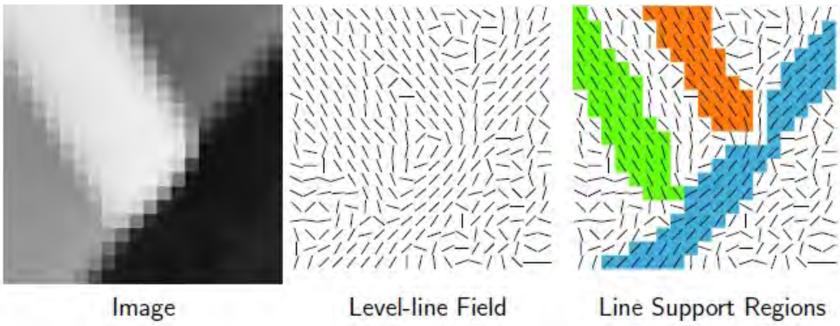


FIGURE 2.8: Level field for image patch. Highlighted regions are support regions indicating possible line segments [10].

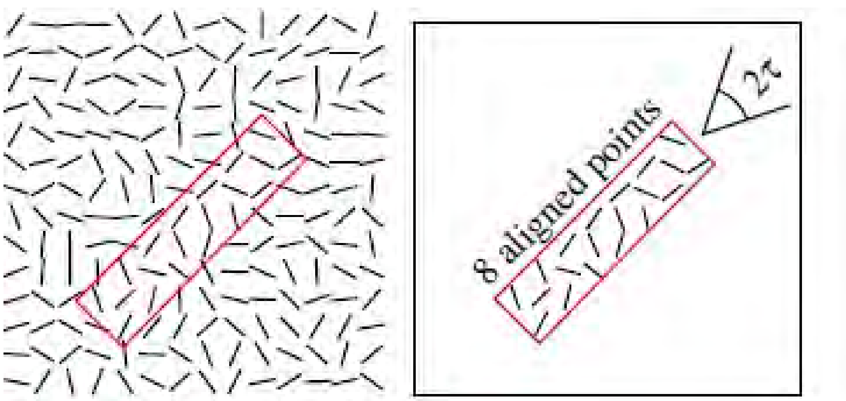


FIGURE 2.9: A line support region [10].

changing the internal and fixed parameters would change the nature of the algorithm. These parameters are image scale, Gaussian sigma value, quantization error threshold, gradient angle tolerance and detection threshold, and the default settings provided by [10] are applicable to all images.

The output of the Line Segment Detector (LSD) will be different if different image scales are used [10]. A lower-scaled image disregards content which may enrich line structures and so fewer lines will be detected than when a higher scale setting is used. This kind of behaviour can also be anticipated because the number of lines imaged when a camera observes a scene from afar differs from when a close-up shot is taken. Considering these effects, the LSD algorithm automatically adapts to these changes if only the scale parameter is changed [10].

2.6 Structure from motion

A structure from motion problem is a multiview geometry problem that involves finding the pose for every camera while simultaneously building a 3D point cloud of the world. Features in the image space are found (often via SIFT) across all available views and matched based on descriptors that describe these features. Reconstructions from valid matches, together with refining steps allow for camera models and a 3D point cloud. Structure from motion is popular in the field of robotic vision and is useful for both navigation and reconstruction purposes. A structure from motion pipeline has even been implemented in crack detection for power components [20], whereby a scaled scene helped quantify the length of defects. The two main structure from motion pipelines are global and sequential pipelines with sequential (or incremental) strategies often at greater risk of being subjected to error drift [33] (global pipelines intend to distribute error residuals evenly).

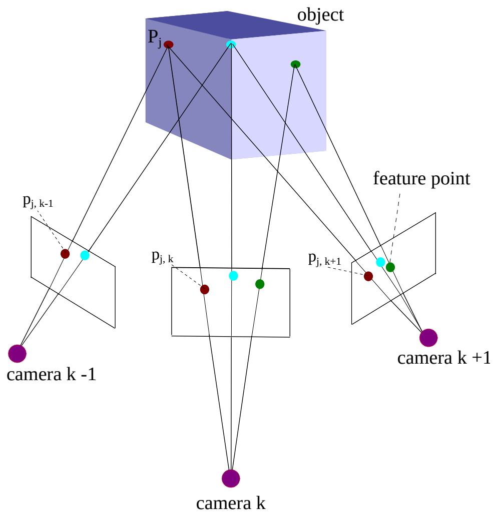


FIGURE 2.10: Conceptual diagram for multiview geometry.

2.6.1 Sequential structure from motion

Sequential structure from motion (SfM) begins with an initial reconstruction that uses two views. With every new view that is then added, a pose for the new camera may be recovered by solving the exterior orientation problem (see Section 2.6.2). New feature points from incoming images in a sequence are allowed to contribute to the 3D point cloud [22] and finally, a refining step (bundle adjustment) is performed (this means solving a non-linear optimization problem) [14] to minimize error propagation and drift across all cameras. Conventional pipelines also use RANSAC [6] techniques which are algorithms that have thresholds to discriminate between inliers and outliers when determining camera pose. These values are often set empirically and remain fixed as in popular SfM packages like Bundler [42].

Unlike most sequential structure from motion pipelines, the approach introduced by [33] does not depend on fixed thresholds. The values are adaptive whilst allowing for a more accurate reconstruction. No user-intervention is required when setting thresholds during RANSAC operations because [33] uses an *a contrario* method when establishing feature correspondence, as well as during camera pose estimation. This *a contrario* approach is based on the Helmholtz principle [4] which states that it is unlikely for a structured configuration to be visible by chance. False alarms, in this context, would be models that indeed arise out of chance. As a result, the algorithm doesn't follow conventional methods of maximizing the number of inliers. With [33], minimizing the number of false alarms during RANSAC operations is the objective.

2.6.2 Exterior orientation

When 3D points have corresponding 2D representations, a calibrated camera's pose and position may be desired as is the case in a structure from motion pipeline. Some popular algorithms that address this is the POSIT algorithm [2]. This algorithm approximates a perspective projection using scaled orthographic projection and is able to extract camera extrinics by solving a linear problem using iterative steps to improve results. This algorithm falls into the class of Perspective-n-Point (PnP) problems that use n point correspondences to determine camera pose and position [40]. Three points form the smallest subset that can yield a solution, provided that a fourth point is used to disambiguate amongst solutions. A state-of-the-art solution to the P3P (three points) problem is one that exploits the cosine rule and uses intermediate frames for both the camera and the world [22]. It is recommended that PnP algorithms are run after RANSAC [6] to remove outliers and minimise false correspondences.

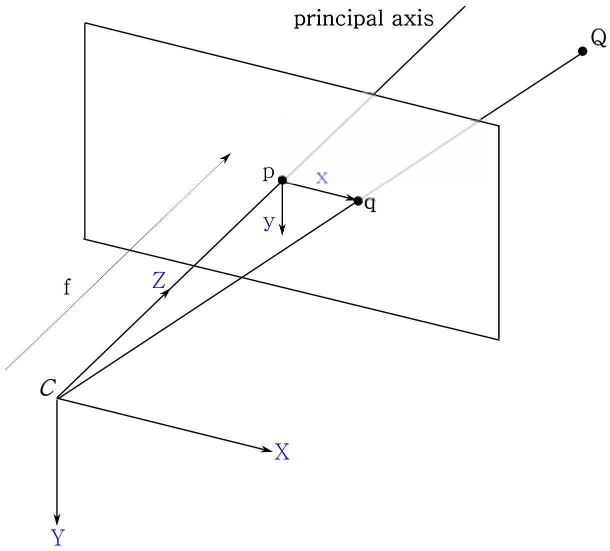
Chapter 3

Camera model and projective geometry

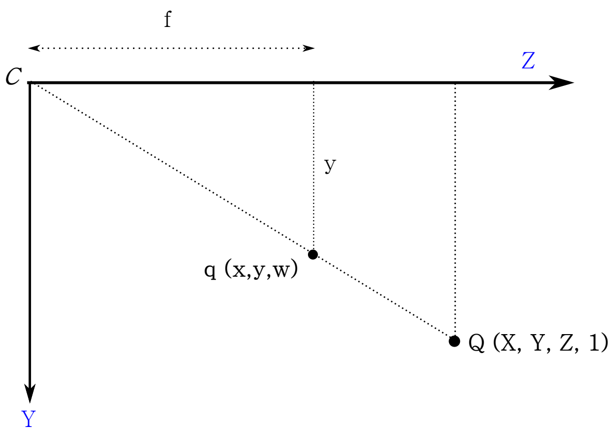
This chapter introduces the fundamentals of the pinhole camera model that is used for 3D reconstructions. The link between epipolar geometry and the essential matrix is also studied, using two views. Finally, the use of bundle adjustment is acknowledged towards the end of this chapter as it is responsible for refining a set of reconstructed cameras. The reason for presenting these areas of projective geometry is because they are used in the implementation of a tower reconstruction strategy discussed in Chapter 4.

3.1 Pinhole camera model

The pinhole camera is based on a thin-lens model [7]. This model can be used to dictate the projection of a 3D point onto its 2D equivalent on the image plane. To break down the construct of a single camera, Figure 3.1 is provided.



(A) A 3D projection of a point onto a 2D image plane.



(B) Side view of 3D-2D projection.

FIGURE 3.1: The image plane, camera centre and a world point form similar triangles.

Figure 3.1 shows coordinates C which is the camera centre in $\mathbb{R}^3$. The focal length for this pinhole model, f, is the distance between the camera centre (or optical centre) and its image plane. The principal axis (positive Z axis in this case) penetrates this image plane at a principal point p. Between a 3D component of Q, Y, and its image representative, y, there is a distinct and non-linear relationship,

$$\begin{aligned} \frac{Y}{Z} &= \frac{y}{f} \\ y &= f\frac{Y}{Z} \end{aligned} \tag{3.1}$$

The same similar triangles approach, shown in Figure 3.1, may be used to deduce the x image coordinate.

Using homogeneous coordinates to convey these projections as linear operations, a 3D point $\mathbf{X} = (X, Y, Z, 1)^T$ is projected onto a 2D image point $\mathbf{x} = (x, y, \lambda)$ as follows,

$$\mathbf{x} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = K_f \Pi_0 \mathbf{X} \tag{3.2}$$

where

$$K_f = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \Pi_0 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{3.3}$$

Let Π_0 be a canonical matrix while K_f is a calibration matrix modelling focal lengths. This calibration matrix cannot be influenced by camera pose and position. A practical calibration matrix models parameters (camera intrinsics) such as focal lengths (in x and y direction) as well as a principal point that may not be at the origin. Skew is an additional intrinsic parameter to compensate for when the image plane axes are not truly orthogonal, but this is often ignored with modern cameras. It will not be represented in Equation 3.3 for this work. In addition to these parameters, manufacturing errors in lenses cannot be avoided as this leads to image distortion [48]. Therefore all images considered in this work are undistorted.

3.1.1 Intrinsics

A more insightful explanation of camera intrinsics can be provided by rewriting Equation 3.2 which represents a mapping of 3D points onto 2D image points. Starting with

$$\lambda \mathbf{x} = K_f \Pi_0 \mathbf{X}, \quad \lambda > 0, \tag{3.4}$$

where λ is some unknown scale value, if the 2D origin on the image plane is not coincident with principal point p, these would affect the similar triangles relationship as follows:

$$\begin{aligned} x &= f\frac{X}{Z} + p_x \\ y &= f\frac{Y}{Z} + p_y. \end{aligned} \tag{3.5}$$

Furthermore, if the resolution of the image (pixels/mm) is not the same in the horizontal (s_x) and vertical (s_y) directions, these would need to be added to Equation 3.5 i.e.

$$\begin{aligned} x &= s_x(f\frac{X}{Z} + p_x) \\ y &= s_y(f\frac{Y}{Z} + p_y). \end{aligned} \tag{3.6}$$

Combining these observations, the mapping of a 3D coordinate $\mathbf{X}$ to pixel coordinates becomes

$$\lambda \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{bmatrix} s_x & 0 & p_x \\ 0 & s_y & p_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}. \tag{3.7}$$

Using a more compact form, Equation 3.7 can be written as

$$\lambda \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = K_s K_f \Pi_0 \mathbf{X}. \tag{3.8}$$

Both K_s and K_f contain internal parameters and combining them leads to a calibration matrix that is appropriate for cameras today:

$$K = \begin{bmatrix} f_x & 0 & p_x \\ 0 & f_y & p_y \\ 0 & 0 & 1 \end{bmatrix}. \tag{3.9}$$

This K is what will be used for the rest of the report.

3.1.2 Extrinsics

It is important to note that thus far, the camera model was assumed to be positioned at the world origin making the 3D to 2D projection straightforward. The camera has its own reference frame that may be different to that of a rigid target object. The 3D point $\mathbf{X}$ used in Equation 3.2 is in reference to the camera coordinate system. If the world coordinate system does not align with that of the camera's origin and orientation, a rotation $\mathbf{R}$ and translation $\mathbf{t}$ must be performed before doing any mapping from 3D to 2D. In Euclidean coordinates, this transformation on a world point, $\mathbf{X}_w$, is expressed as

$$\mathbf{X}_c = \mathbf{X}_w \mathbf{R} + \mathbf{t}, \tag{3.10}$$

where $\mathbf{X}_c$ is a 3D point in the camera reference frame.

When a camera frame and world coordinate system are not aligned, parameters $\mathbf{R}$ (for camera pose) and $\mathbf{t}$ (for camera position) in Equation 3.10 are the extrinsic parameters that also form part of the pinhole model. Camera extrinsics are not influenced by camera intrinsics and vice versa. Combing intrinsics and extrinsics leads to a 3×4 matrix that is the pinhole camera $\mathbf{P}$:

$$\mathbf{P} = K[\mathbf{R}|\mathbf{t}] \tag{3.11}$$

which encompasses the 3D to 2D projection for homogeneous points:

$$\mathbf{x} = \mathbf{P}\mathbf{X}_w \tag{3.12}$$

where $\mathbf{X}_w$ is $(X, Y, Z, 1)^T$ and $\mathbf{x} = (x, y, 1)$. All quantities expressed in Equation 3.12 are homogeneous meaning that, for example, $s\mathbf{P}$ is the same modelled camera as $\mathbf{P}$. It is for this reason λ from Equation 3.7 is dropped.

3.1.3 Optical centre and principal axis vector

The optical centre and pricipal axis vector are required for the reconstruction process explained in Chapter 4 . Obtaining the optical axis is done by finding the right null-vector of $\mathbf{P}$ such that $\mathbf{P}C = 0$ [14].

To find the forward principal axis vector (aligned in front of the image plane), it is first noted that the camera matrix can be written as $\mathbf{P} = \left[\tilde{M} \mid \mathbf{p_4}\right]$, where $\tilde{M}$ is the first 3 columns of the camera matrix. With $\mathbf{m}^{3T}$ being the third row of $\tilde{M}$, and because $\det(\mathbf{R})$ is always positive, this forward-pointing vector, $\mathbf{v}$, can be achieved as follows:

$$\mathbf{v} = \det(\tilde{M})\mathbf{m^3}. \tag{3.13}$$

3.2 Two-view geometry

For reconstruction of an object from multiple views, it is better to start understanding a two-view problem. Recovering the second camera's extrinsics is possible using epipolar geometry as well as sufficient points that are in correspondence. For two pinhole camera models that capture the same scene, but from different perspectives, a setup is shown in Figure 3.2.

In Figure 3.2, there are two image planes in the presence of a 3D point $\mathbf{X}$ and a baseline connects the two camera centres. (Let the left camera, C, be denoted as the first camera while the right camera C' acts as the second camera). The projection of point, $\mathbf{X}$, onto the two image planes results in homogeneous image points $\mathbf{x}$ and $\mathbf{x}'$. Epipoles are the mapping of one camera centre

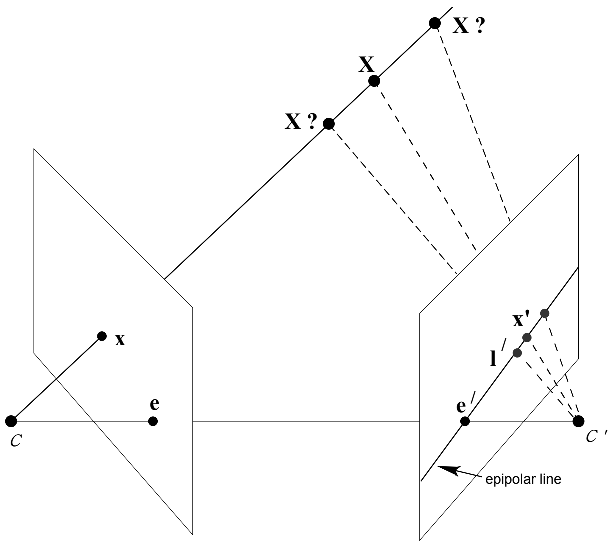


FIGURE 3.2: Epipolar plane containing the camera centres $C$and C', 3D point $\mathbf{X}$ as well as image points $\mathbf{x}$ and $\mathbf{x}'$
[14].

into the other camera's image. In other words, $\mathbf{e}'$ is the image of camera C. This is also shown in Figure 3.2, together with an epipolar line, $\mathbf{l}'$, that is formed between the observed image point $\mathbf{x}'$ and the epipole. This epipolar line can be thought of as the image of the line-of-sight between C and $\mathbf{X}$. For an observed point $\mathbf{x}$ on the image plane of C, the corresponding viewpoint on the image plane of C' lies somewhere along $\mathbf{l}'$ i.e. a point in one view is mapped to a line in the second view. Two vectors $C\mathbf{x}$ and $C'\mathbf{x}'$, as well as the baseline, form an epipolar plane. A different plane – with the same baseline – would be present if another 3D point was considered. This epipolar plane creates a constraint, which can be manipulated when a second camera's extrinsics needs to be calculated.

3.2.1 Epipolar constraint and the Essential Matrix

By considering the baseline in Figure 3.2 as t_b, and in addition with two vectors, $C\mathbf{x}$ and $C'\mathbf{x}'$, that form an epipolar plane, the following can be expressed:

$$\mathbf{x} \,.\, (t_b \times \mathbf{x}') = 0. \tag{3.14}$$

However, if $\mathbf{x}'$ can be written with reference to the origin of camera C then

$$\mathbf{x} \,.\, (t_b \times \mathbf{R}\mathbf{x}') = 0. \tag{3.15}$$

Writing t_b as a skew symmetric matrix i.e. $[t]_x$,

$$\mathbf{x}^T \, E\mathbf{x}' = 0 \tag{3.16}$$

where E is the essential matrix that describes the rotation and translation between the two views as $[t]_x \mathbf{R}$. There is a limitation, however. Without some external measurement of the real world, there is no way to recover the true length of the baseline. Supposing that the baseline between cameras C and C' were halved and the 3D point was twice the distance away from the cameras, the same images would appear. Therefore, the only useful information about the translation recovered is the *direction*.

3.2.2 Eight point algorithm

With sufficient point correspondences between two images, it is possible to estimate camera extrinsics between two cameras in the form of an essential matrix. This is where the eight point algorithm [25] is applied. Having a two-view initial reconstruction in a structure from motion pipeline is necessary before all subsequent images are added. For two views, one of the camera extrinsics may purposefully be set at the world origin. This would imply that the second camera's position is with respect to that of the first. To obtain the essential matrix, 8 image points need to be matched across both images. These may be found by specialised feature detection techniques like SIFT.

Using the homogeneous linear equation from Equation 3.16 consider points $\mathbf{y} = \begin{pmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \\ \mathbf{y}_3 \end{pmatrix}$ and $\mathbf{y}' = \begin{pmatrix} \mathbf{y}'_1 \\ \mathbf{y}'_2 \\ \mathbf{y}'_3 \end{pmatrix}$. Letting the essential matrix be $E = \begin{bmatrix} e_{11} & e_{12} & e_{13} \\ e_{21} & e_{22} & e_{23} \\ e_{31} & e_{32} & e_{33} \end{bmatrix}$ and expanding results in:

$$\begin{bmatrix} (\mathbf{y}'_1 e_{11} + \mathbf{y}'_2 e_{21} + e_{31}) & (\mathbf{y}'_1 e_{12} + \mathbf{y}'_2 e_{22} + e_{32}) & (\mathbf{y}'_1 e_{13} + \mathbf{y}'_2 e_{23} + e_{33}) \end{bmatrix} \begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \\ \mathbf{y}_3 \end{bmatrix} = 0 \quad (3.17)$$

or:

$$\mathbf{y}_1(\mathbf{y}'_1 e_{11} + \mathbf{y}'_2 e_{21} + e_{31}) + \mathbf{y}_2(\mathbf{y}'_1 e_{12} + \mathbf{y}'_2 e_{22} + e_{32}) + (\mathbf{y}'_1 e_{13} + \mathbf{y}'_2 e_{23} + e_{33}) = 0. \quad (3.18)$$

A more compact expression is:

$$e \cdot \tilde{\mathbf{y}} = 0 \quad (3.19)$$

with $\tilde{\mathbf{y}} = \begin{pmatrix} \mathbf{y}'_1\mathbf{y}_1 & \mathbf{y}'_1\mathbf{y}_2 & \mathbf{y}'_1 & \mathbf{y}_1\mathbf{y}'_2 & \mathbf{y}_2\mathbf{y}'_2 & \mathbf{y}'_2 & \mathbf{y}_1 & \mathbf{y}_2 & 1 \end{pmatrix}$ and e containing all 9 elements of the essential matrix.

This illustrates how between two images, 8 points are required to solve for the essential matrix E (there are 9 terms in the expression but the 9^{th} element represents scale and translation is conveniently set to be a unit vector when solving for E).

However, acquiring E thus far presents a four-fold ambiguity in the correct solution. This is because the essential matrix E does not consider sign as part of the solution [14]. A simple check to see which solution is desired requires testing the set of extrinsics which presents the scene in front of both cameras.

To summarize the usefulness of the essential matrix, a rotation matrix, $\mathbf{R}$, and unit direction vector representing translation, $\mathbf{t}$, can be obtained from E. The true length of $\mathbf{t}$ can only be obtained once a true measurement in the world is taken and applied to the two-view reconstruction.

3.3 Multi-view geometry and bundle adjustment

When more views are present, and subsequently, more camera extrinsics are desired, a structure from motion problem results. This was discussed in Section 2.6.1 which starts with a two view reconstruction before adding more views sequentially to provide camera poses and a sparse 3D model in one coordinate system. An incremental structure from motion pipeline will have new images undergo the process of having features detected (SIFT) and an initial camera pose suggested by RANSAC. The exterior orientation problem estimates the camera pose and bundle adjustment refines all 3D points and camera poses obtained thus far. The P3P algorithm [22] is provided by the openMVG library and is the suggested method for camera resectioning.

3.3.1 Bundle adjustment

In the context of incremental structure from motion, a refining step is necessary whenever a new image is added i.e. a refinement on the cameras (and 3D model). Where i refers to the camera number and j the image and/or world point, bundle adjustment models the discrepancy between observed $\tilde{\mathbf{x}}_j^i$ and its predicted image point as follows:

$$Err = \tilde{\mathbf{x}}_j^i - K(\mathbf{R}_i\mathbf{X}_j + \mathbf{t}_i). \tag{3.20}$$

The deviation of the observed image point, $\tilde{\mathbf{x}}_j^i$, is based on a Gaussian distribution around the true image point $\mathbf{x}_j^i$. This assumption (other probabilistic distributions do exist, but a Gaussian type is more common) means that a maximum likelihood solution to joint camera motion and structure is provided [14].

The nonlinear cost function (based on reprojection error) in a bundle adjustment algorithm is expressed as

$$\min_{\hat{\mathbf{P}}^i, \hat{\mathbf{X}}_j} \sum_{ij} d(\hat{\mathbf{P}}^i \hat{\mathbf{X}}_j, \tilde{\mathbf{x}}_j^i)^2. \tag{3.21}$$

In Equation 3.21, the parameters solved for are best-fit solutions for camera motions $\mathbf{P}^i$, with $i = 1...m$ and structures $\mathbf{X}_j$ where $j = 1...n$.

Chapter 4

Tower reconstruction from images

Transmission towers present a source of geometrically-rich features that can be manipulated for modelling. Towers lack texture, but consists of several line segments (beams) that are configured in an arranged construct. These individual line segments make up the wiry nature of the tower. This observation is exploited in the algorithm designed by [16]. The fact that the tower is generated from 2D images inspires this study to adopt this approach.

4.1 Line based 3D reconstruction of a tower

The objective of [16] is to build a wiry model from several 2D images where the camera model for all frames is known *a priori*. This knowledge is acquired by solving the sparse structure-from-motion problem [19, 47], as well as a SIFT descriptor-matching algorithm to establish correspondences across multiple views. The scale of the target was worked out by measuring known marker distances. Having known information about the individual frames involves knowing the camera pose and position as well as camera internal parameters: focal length and principal point. These pinhole camera concepts were explained in Chapter 3. From [16], three important phases of the reconstruction algorithm can be isolated, namely: 3D hypothesis generation, 3D line scoring, and line clustering. It is after the clustering process that a refined 3D model of a tower reconstruction is obtained. For the experiment in [16], the camera moved around the tower and 106 images were captured.

4.1.1 Hypothesis generation

In image I_i, if the Line Segment Detector (LSD) is applied, the result is an array listing the finite line segments' start and end points. Let the start and end points of a particular line segment, l_i, be 2D coordinates p and q, respectively. Epipolar geometry then aims to find the same line

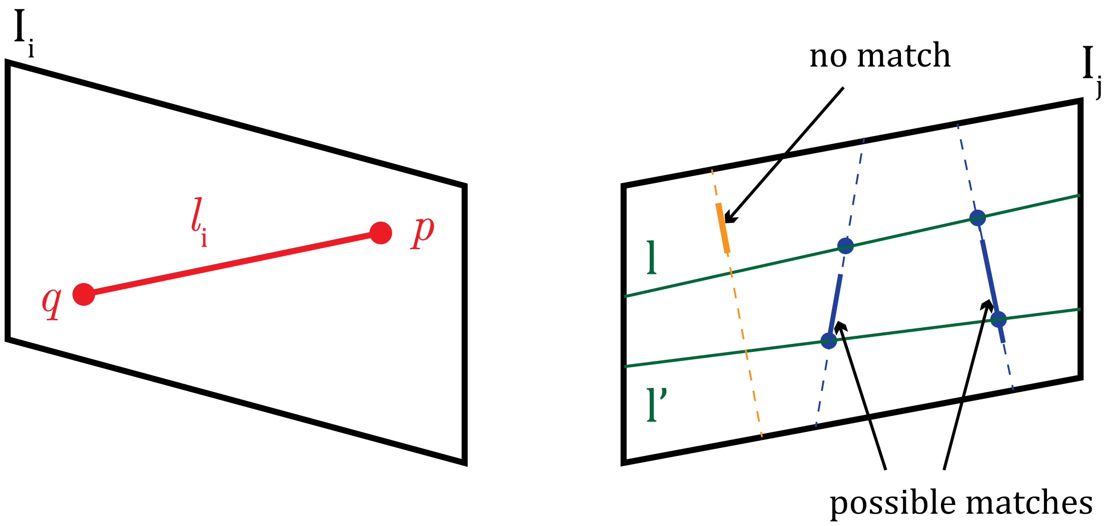


FIGURE 4.1: Hypothesis generation based on epipolar geometry [16].

segment in another image I_j by exploiting the fact that a point in one image has a search space along an epipolar line in another image. To further illustrate, Figure 4.1 is provided.

In Figure 4.1, the conceptual (green) epipolar lines $\mathbf{l}$ and $\mathbf{l}'$ are established in image I_j using epipolar geometry, while line segments (for example, l_i) depicted in Figure 4.1 are found by the LSD algorithm [10]. Examples of line segments detected in image I_j are the two blue lines and the single orange line segment. Line l_i's equivalent line in image I_j would lie between both epipolar lines and ideally start and end on $\mathbf{l}$ and $\mathbf{l}'$. However, more than one candidate line may be found by the LSD between $\mathbf{l}$ and $\mathbf{l}'$. Furthermore, the start and end points of these segments may also fall short of the epipolar lines, which is why all lines between $\mathbf{l}$ and $\mathbf{l}'$ (the two blue lines in this example) are potential matches for line l_i. The notion is that for multiple 2D hypotheses there can be at most one true match for line l_i. Another possibility is that the 2D candidates are false positives and are not related to l_i. At this stage, no indication is provided by the algorithm about which lines may be accepted. Therefore, all 2D candidates inside the epipolar region are considered as hypotheses. Meanwhile, the orange line segment in image I_j cannot be considered a 2D candidate. It is outside the bounds of the epipolar region. Overall, this example illustrates how more than one hypothesis may be detected in 2D. After acquiring all 2D hypotheses (for all lines in all images), the algorithm triangulates these candidates to 3D and performs a scoring operation to discriminate between good hypotheses and false line segments (hypotheses in 2D that do not intersect the epipolar region are extrapolated until they do so). At the end of the hypotheses generation step, the result is a set of 3D lines S.

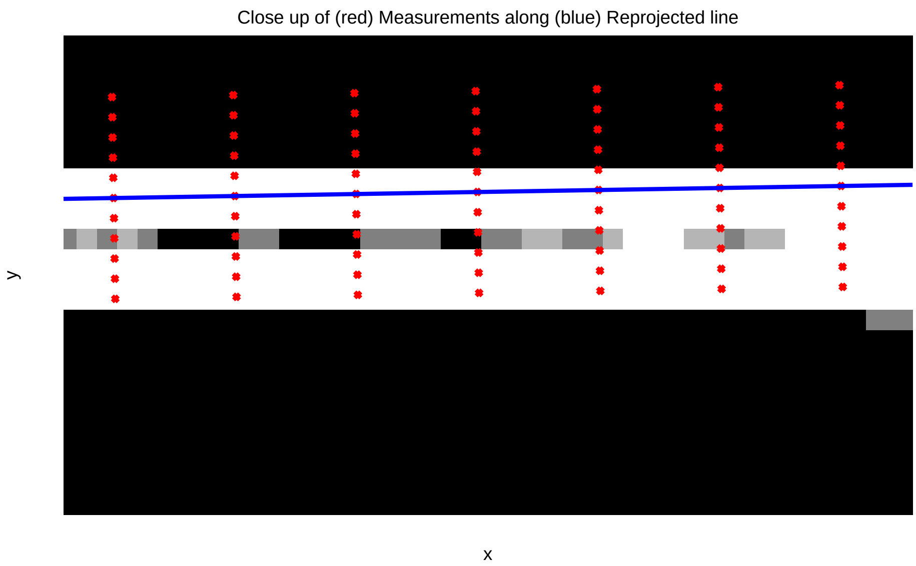


FIGURE 4.2: The image-based scoring mechanism generated in MATLAB.

4.1.2 Hypotheses scoring

Scoring iteratively decides which 3D lines in S are spurious and which lines may be part of the final model. The scoring stage rates the quality of these candidate segments but requires the construction of image neighbourhoods. To understand what constitutes an image neighbourhood, consider an image I_i and an image I_j. A 'neighbour' is that image whose camera centre is at distance – at most d_c metres – away from the camera centre of image I_i. This parameter is a physical measure in the world frame. In addition, the absolute difference in viewing angle between I_i and I_j must be within a threshold, d_{ang} degrees. If these two conditions are met, image I_j qualifies as a neighbour of image I_i. A fixed number of images that are neighbours to image I_i result a neighbourhood $N(I_i)$. The size of a neighbourhood is determined by the number of neighbours present, but [16] capped their neighbourhood size to 20 due to available computational power. Furthermore, thresholds, d_c and d_{ang}, were set to 30 m and 50° respectively and these values were based on *a priori* knowledge of the scale of the scene [16].

Image neighbourhoods are constructed because the set of lines S needs to be backprojected into these groups which should view the target similarly (given parameters d_c and d_{ang}). During the scoring process, when a particular image is used to assess the backprojected lines S, it is in fact the image gradient that is used. At discrete locations within the gradient image, the backprojected line is compared to the image gradient to see if there is an overlap (a well-reprojected line segment). Figure 4.2 demonstrates this idea.

In Figure 4.2, red measurement points (M) are the discrete locations in an image, used to establish a gradient-based score. These points are oriented perpendicularly to the backprojected line (blue), and are spaced a fixed number of pixels across the line. The score for a particular 3D line is normalized so that the same line is allowed to change in length (as expected in different views). Normalization for a particular 3D line is therefore done across all images within a particular neighbourhood, and eventually, across all neighbourhoods. The score or measure for a particular line L_m in set S is expressed as

$$s(L_m) = \frac{1}{|N(I_i)|} \sum_{I \in N(I_i)} \sum_{\mathbf{x} \in M(I)} \frac{||VI(\mathbf{x})||}{|M(I)|} e^{-\left(\frac{\lambda.dist(\mathbf{x},L_m)}{2.dist_{max}(L_m)}\right)^2}. \tag{4.1}$$

In order to understand Equation 4.1, consider a set of pixels that span L_m and are arranged in 'rungs' (see Figure 4.2) perpendicular to L_m. Each rung is 11 pixels long and is intermittent (occurs after every 5 pixels) along L_m. A rung on L_m has five discrete points on either side of L_m. These pixels are the measurement pixels, M. The image gradient, $VI(x)$, is measured at position $\mathbf{x}$ while $dist(\mathbf{x}, L_m)$ is the Euclidean distance between the measurement point $\mathbf{x}$ and L_m. The $dist_{max}(L_m)$ term is the maximum possible distance between $\mathbf{x}$ and L_m, and the $e^{-(\frac{\lambda.dist(\mathbf{x},L_m)}{2.dist_{max}(L_m)})^2}$ term weights the score at location $\mathbf{x}$ differently, such that $\mathbf{x}$ furthest away from L_m is worth less than a pixel directly on top of L_m. In Figure 4.2, L_m projects well onto the image gradient pixels, possibly suggesting a true line segment.

After scoring all the 3D lines in S, high scoring lines are expected to be lines supported by most of the 2D views. Such lines would conform to the structure of the target. Unwanted, noisy lines should have low scores because they would not be consistently supported in the majority of the image neighbourhoods. After scoring, the clustering process follows.

4.1.3 Clustering

The lines in S are then ordered by score (from highest to lowest) to facilitate the clustering process. For each line, L_m – starting with the line that has the highest score – a cylinder in 3D is constructed having radius r metres. The central axis of the cylinder is the line L_m expanded by 10%. The decision to expand L_m by this amount was not explained in [16], but suited the reconstruction process based on a set of estimated camera matrices. All 3D lines $L_n, n \neq m$ are then checked to see if both their endpoints fall within this cylinder. If they do, they are clustered into cylinder L_m. If the number of lines in a cylinder, including the central axis, is denoted as H, and $H \geq 3$, a successful cluster is formed. If H falls short of the minimum number of lines expected (H_{min}), L_m is discarded. Lines that were consumed into a cluster are removed from S so that low-scoring lines that are already part of a cluster do not create cylinders themselves. Figure 4.3 is provided to help visualize this 3D clustering cylinder.

Notice that the central axis and blue lines in Figure 4.3 belong to a cluster. The red line has an endpoint outside the cylinder and this results in it being excluded from this cluster. In Figure 4.3, the cylinder has three lines which suggests a successful cluster (if $H_{min} = 3$).

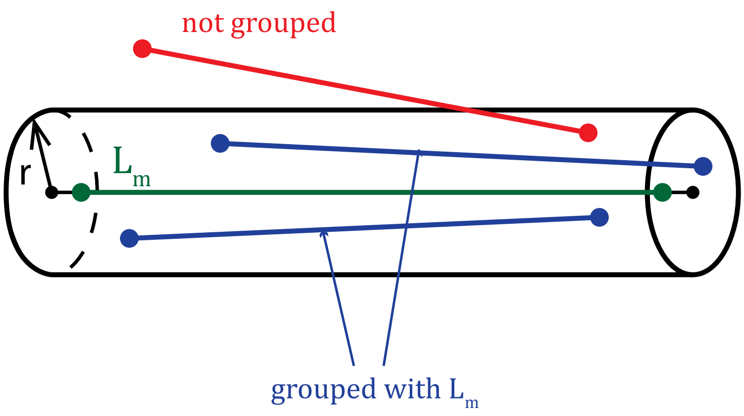
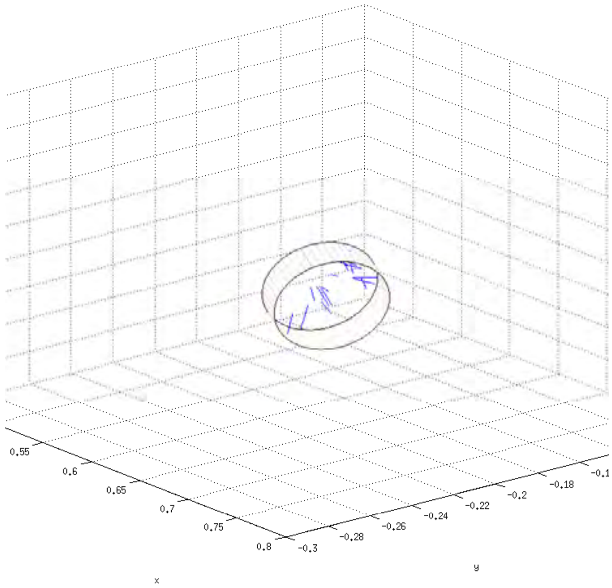
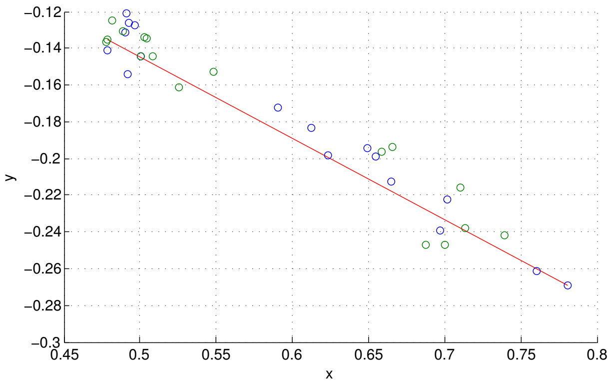


FIGURE 4.3: Cylinder with lines and an outlier [16].

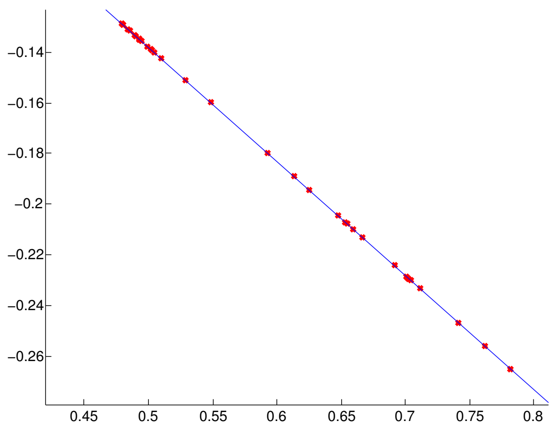
After achieving all possible clusters, a generalised line segment with length and orientation is derived for each cylinder. This is so that all line segments in their respective groups can collapse onto a single representative line segment. To obtain the direction and length of a generalised line segment, the centre of gravity in the cylinder is determined, using the cluster-members' endpoints. Singular value decomposition on these endpoints is performed and the largest eigenvalue corresponds to an eigenvector i.e. the direction of this representative vector. All endpoints in the cylinder are then projected onto this vector, and the two outermost points on the direction vector span a distance that determines the length of the model line segment. To illustrate how a cylinder with lines collapses to a single line segment, Figure 4.4 is provided.



(A) An example of a cylinder with encapsulated lines.



(B) A cluster with all endpoints.



(C) All endpoints projected onto the direction vector. Length spanned by two-outermost points.

FIGURE 4.4: All lines accepted in a particular cylinder collapse onto a final line segment.

The line endpoints for a particular cluster are shown in Figure 4.4B, where green points are starting points and blue points are end points. Figure 4.4C illustrates the projection of the endpoints onto a vector whose direction will dictate the orientation of this particular model line segment. This is done for all successul clusters. Depicted in Figure 4.5 is a reconstructed transmission tower from [16].

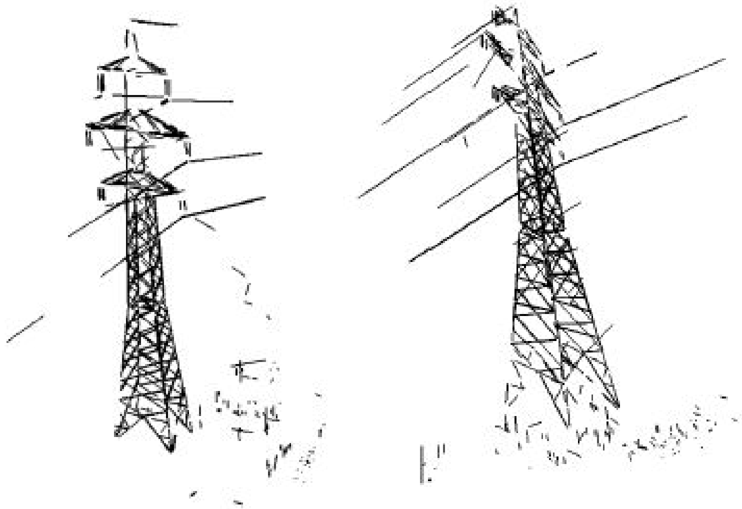


FIGURE 4.5: Final tower reconstructed [16].

Figure 4.5 contains 1000 line segments. Originally, there were 70 000 hypotheses in 3D that were generated from multiple combinations of 2D images (Figure 4.6).

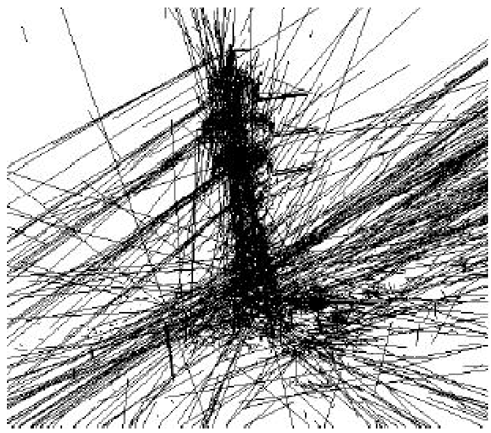


FIGURE 4.6: Line segments (3D) before scoring and clustering [16].

This work will use the method designed by [16] to reconstruct a model tower since it exploits the geometry of the target. Is also uses the fundamentals of projective geometry to hypothesise candidates.

Chapter 5

Application of wiry model reconstruction

This chapter applies the knowledge of recovering multiple camera extrinsics to build a 3D reconstruction of a tower. This two part problem involves using a structure from motion pipeline, as discussed in section 2.6.1, and finally applies the reconstruction algorithm described in Chapter 4. Note that the line reconstruction algorithm explained in the work of [16] is coded from scratch in MATLAB [30] for this research. This wiry reconstruction approach is selected for its ability to manipulate epipolar geometry as well as the wiry construct of a tower. The features (2D lines) required are based on the geometry of the target. Two experiments involve using the reconstruction code on a model tower as well as on the dataset (camera models and 2D) provided by [16].

5.1 Building multiple camera models

Before any tower can be reconstructed, the camera poses need to be recovered. Since the objective is to have a scaled reconstruction, these cameras must be scaled using a real world reference. For the experiment involving a model tower, camera models were established using a sequential SfM pipeline [33] and the openMVG library [35]. The camera used to capture data was a 14 megapixel Panasonic camera [36].

5.1.1 Intrinsics

There were 22 images taken around a balsa wood tower model. These were indexed from 0 to 21 with Figure 5.1 showing index 0. The remaining images used can be found in Appendix A.

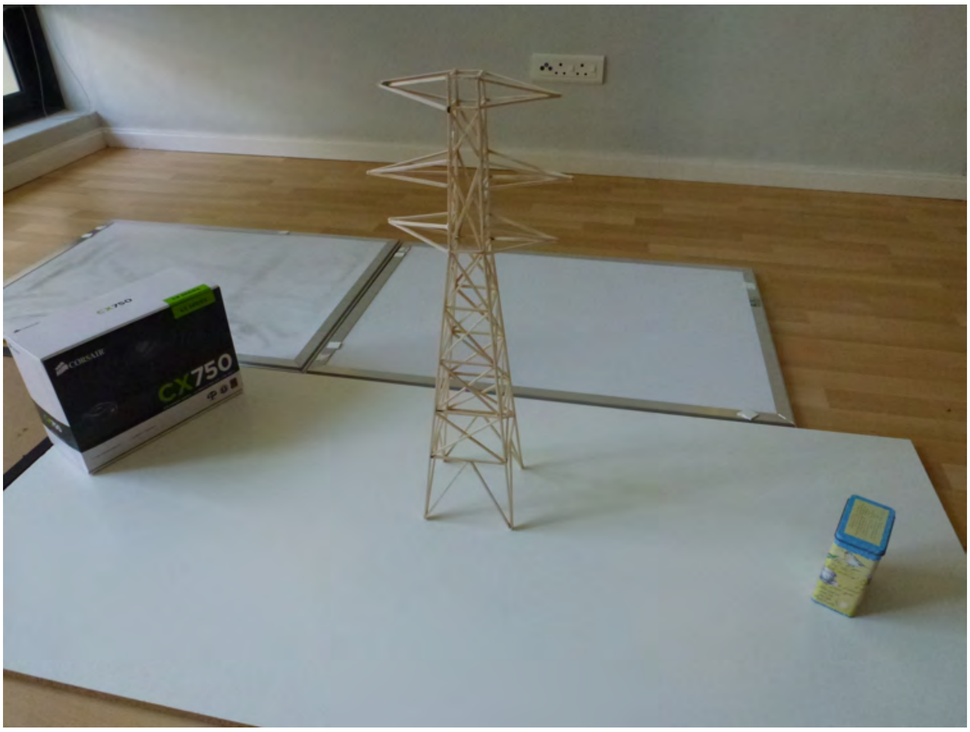


FIGURE 5.1: Tower0.JPG

The intrinsic parameters (focal lengths and optical center) were obtained using MATLAB's calibration app [30]. The image resolution was 2560 $\times$ 1920 pixels and an approximated focal length of 1843.7 pixels was used. This resulted in the following intrinsics matrix for the camera,

$$K = \begin{bmatrix} 1843.7 & 0 & 1288.4 \\ 0 & 1843.7 & 950.3 \\ 0 & 0 & 1 \end{bmatrix} \tag{5.1}$$

whose format subscribes to the intrinsics matrix described in Chapter 3.

Note that the images themselves had to be undistorted, and this was made possible by using the openMVG library. It was imperative that the images were undistorted as an absence of this step would lead to poor line reconstructions.

5.1.2 Camera extrinsics

The camera extrinsics were extracted using a sequential SfM pipeline. These involved several steps that allowed all 22 images participating to have pose estimates. The SIFT algorithm was used to detect feature points across all images and no handpicking of feature point correspondences was required. The initial reconstruction started with the pair of images that had the highest number of corresponding points (image indices 9 and 10) and thus the origin of the world coordinate frame was set to that of image 9's camera centre. Subsequent cameras were added by solving the exterior orientation problem (P3P problem) and performing bundle adjustment. Repeatedly doing this for every new image led to a sequential SfM algorithm. This incremental approach as described in [35] can be broken down into sub-procedures.

SIFT for feature detection

The feature points for images were matched based on their closeness of descriptors. In total over 8000 interest points were found across all images. This method of establishing point correspondences was superior in time and effort than doing the process manually. Table 5.1 is a table of the 10 image pairs that had the strongest point correspondences. The largest pair with the highest matches justified the choice for the two-view reconstruction.

Table 5.1: SIFT point correspondences between 10 image pairs

Image index A	**Image index B**	**No. Matches**
9	10	375
15	16	279
11	12	277
16	17	260
20	21	229
13	14	226
3	4	224
8	9	214
15	17	205
0	21	194

Two view reconstruction

Images 9 and 10 were used to create the two view reconstruction and the final RMSE for the two views is 0.28. Figure 5.2 is provided to show the histogram of residuals for the initial reconstruction.

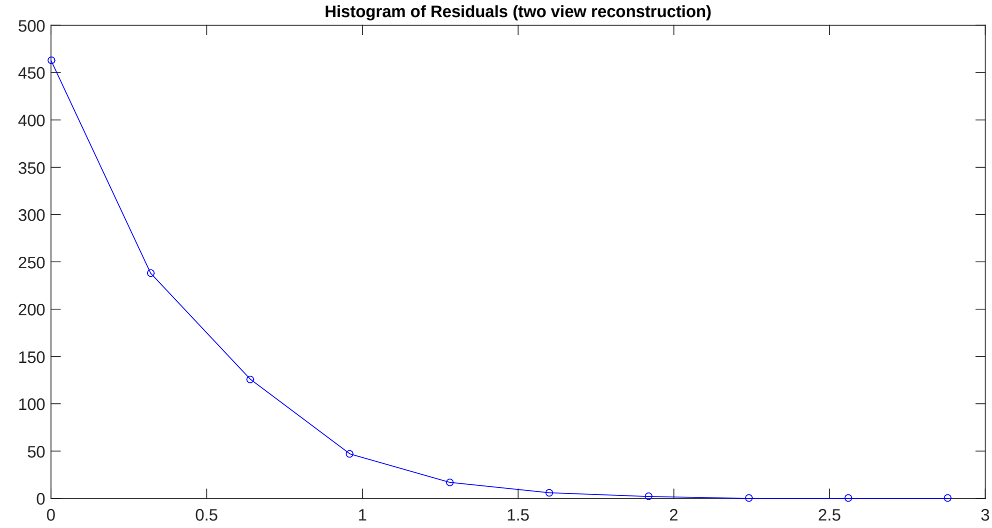


Figure 5.2: Histogram of residuals 9 and 10.

Exterior orientation and others cameras

As part of the SfM pipeline, a camera resectioning step had to be employed for every new image that was added. Camera resectioning or exterior orientation solves for a new camera pose when there are known 3D-2D correspondences between the existing point cloud and the new camera's feature points. To obtain this correspondence, RANSAC is performed. As part of the SfM pipeline, new pose estimates were obtained using the P3P algorithm [22]. Finally, bundle adjustment optimized the point cloud and camera extrinsics involved thus far in the sequence. For all feature points detected, Figure 5.3 shows the reprojection-residual histogram. The largest residual is 3.81. A sub-pixel RMSE value of 0.6 was accepted for all 22 cameras.

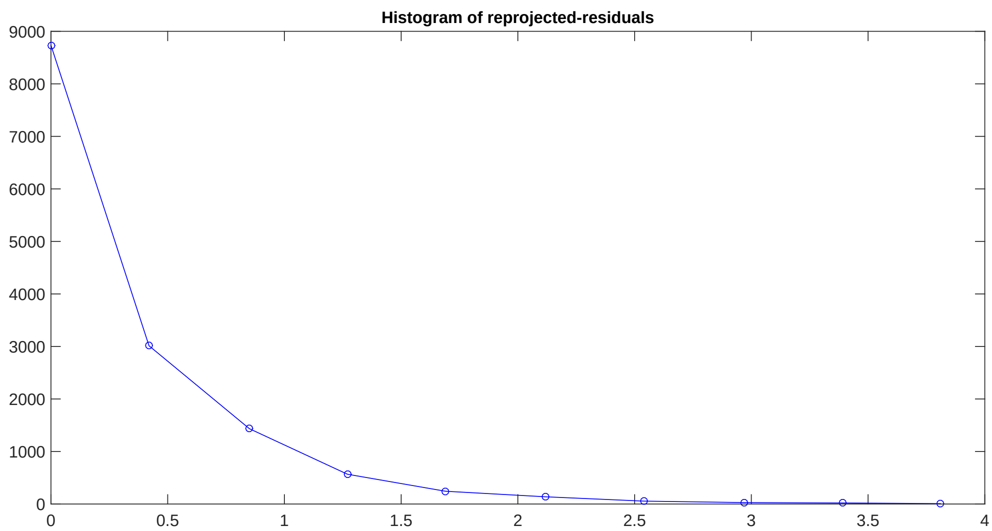
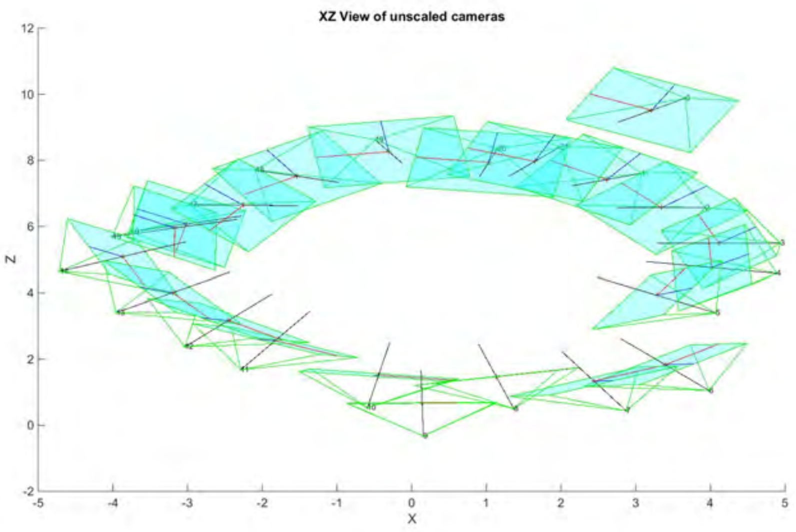
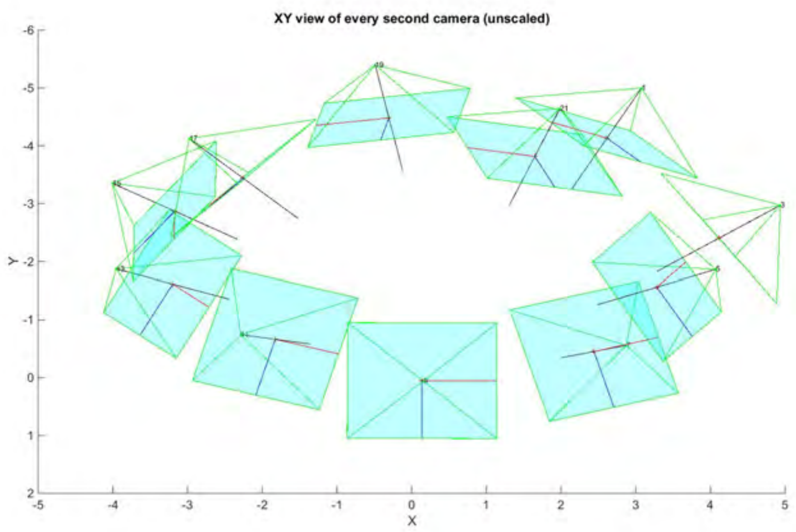


Figure 5.3: Over 8000 points were resectioned with a residual error close to 0.

The final point cloud that resulted from the SfM routine contained 2989 points in 3D and 22 cameras. Not all 8000 feature points detected during the SIFT process were part of the point cloud because as the camera moved; there were interest points that went out of view. It is important to note that the only valuable output from this exercise was to extract camera extrinsics. The point cloud was not consulted for the line reconstruction algorithm. The calibrated cameras and their respective images of the tower formed the subsequent input to the wiry model reconstruction. Every second camera recovered is shown in Figure 5.4. The geometry of the camera positions relative to each other had met expectations. Note that scale correction has not yet been implemented.



(A) XZ view of every second camera.



(B) XY view of every second camera.

FIGURE 5.4: Reconstructed cameras not corrected for scale

Scale correction

A real world measurement had to be taken in order to correct for overall scale. From the point cloud obtained in the SfM pipeline, the length of a model tower segment was used. All camera centres were subsequently corrected for scale.

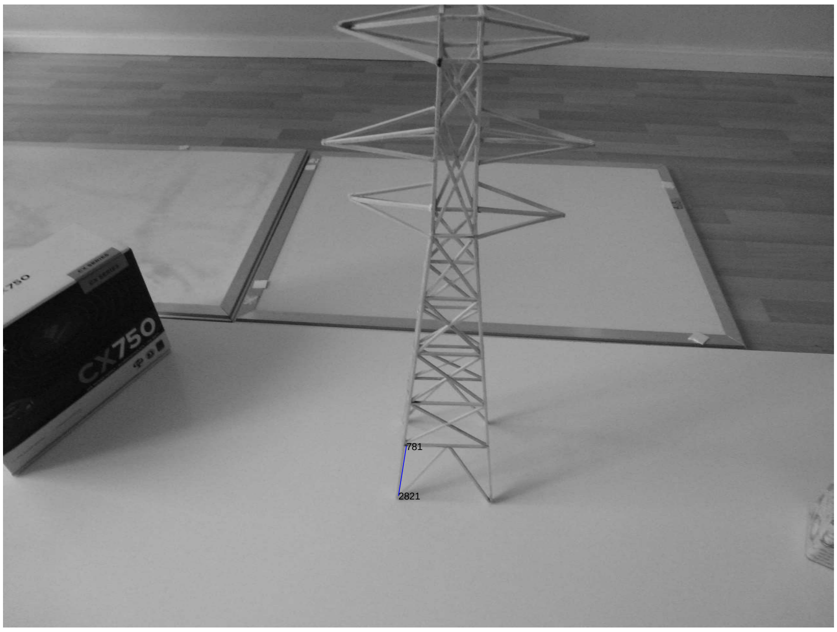


FIGURE 5.5: The highlighted segment was the reference measurement.

In Figure 5.5, the highlighted segment measured 100 mm in real life. The SfM pipeline suggested that the line segment was 0.9 mm and therefore the scale factor was set to 111. Figure 5.6 shows the **XY** view of scaled camera centres (each camera with unit focal length).

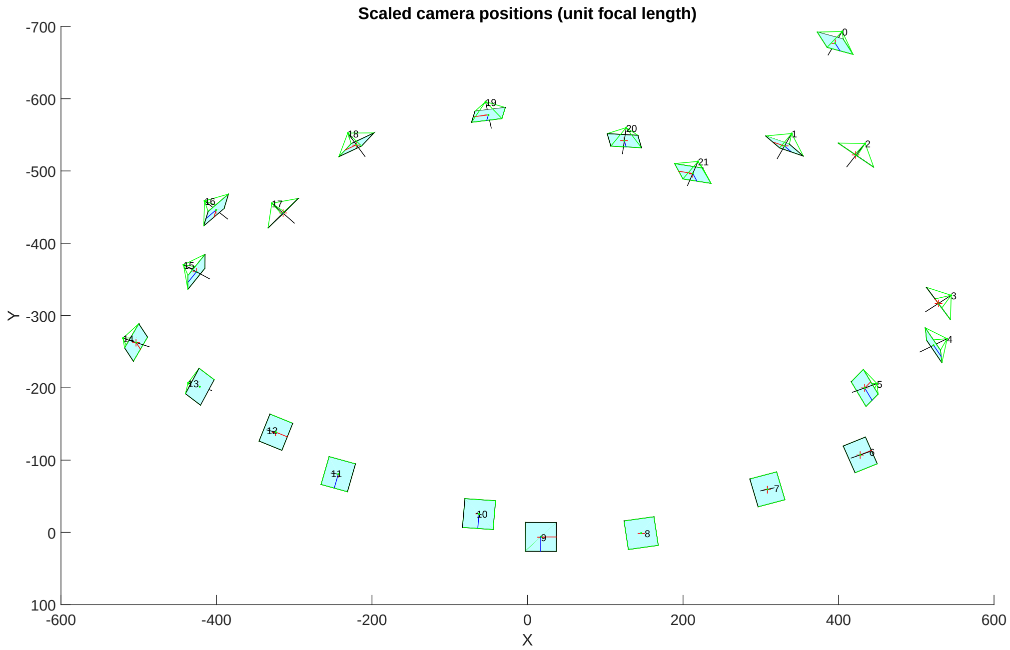


FIGURE 5.6: The **XY** view of scaled cameras.

The span of the camera trajectory in the **X** direction is more than a metre which matches the realistic setup.

5.2 Line Reconstruction of a transmission tower

The line reconstruction algorithm used all 22 images of a balsa tower (about 60 cm in height). In addition to the model tower, the algorithm was also tested on images and cameras provided by Hofer et al. [15]. Undistorted images were used throughout the line reconstruction pipeline, which was coded in MATLAB from scratch.

Hypotheses generation

To generate 3D hypotheses segments, the Line Segment Detector had to operate on all undistorted images to detect 2D images. An example of the output of the LSD algorithm on an image is shown in Figure 5.7.

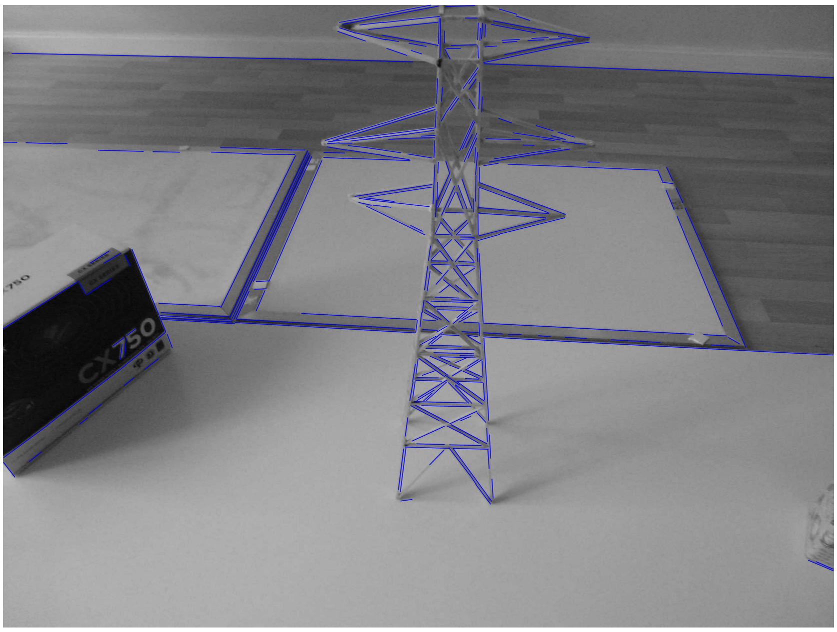


FIGURE 5.7: Line Segment Detector on the first image (Camera 0)

Figure 5.7 shows 2D line segments detected for image 0. All tower lines were not picked up and the wooden floor in the background was also missed, however the skirting board was picked up by the LSD algorithm due to its better contrast quality. A decision was made to continue without changing parameters of the default LSD algorithm. Since multiple images of the tower were being taken, it was deemed permissible to have missing segments as these were assumed to be compensated for in subsequent images.

For every line segment detected in image I_j, hypotheses were found in image I_{j+1}. To illustrate an example of the hypotheses generation, Figure 5.8 is provided. Note that at this stage, the line reconstruction algorithm did not discriminate between good and false hypotheses.

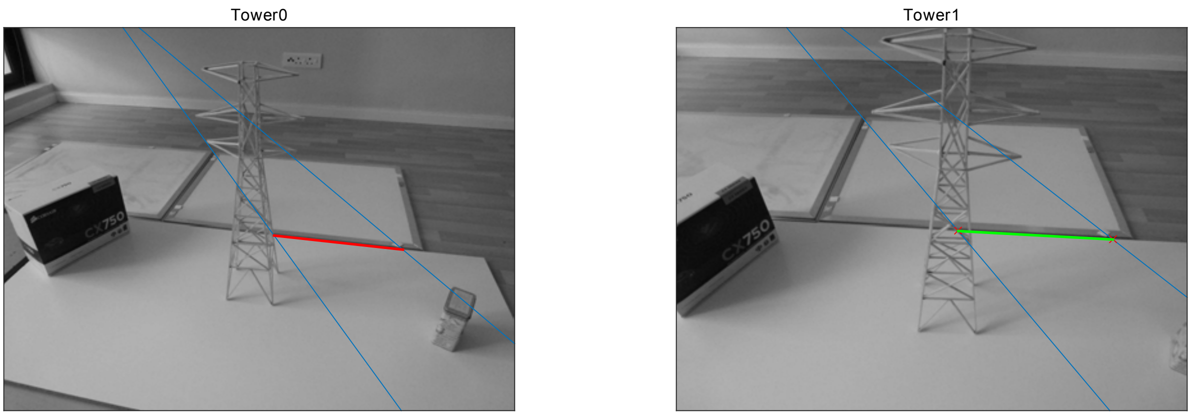


FIGURE 5.8: LSD line (left image) with a valid candidate detected (right image).

The blue epipolar lines in Figure 5.8 pass through the same locations of the physical objects depicted. This was an additional observation that was required to facilitate line reconstruction and would have been affected by improper camera models. A green hypothesis is shown and, in this example, corresponds to the same line segment as the red LSD line. However, the algorithm accepted all possible hypotheses (LSD line segments) that fell within the epipolar region. Another hypothesis is shown in Figure 5.9.

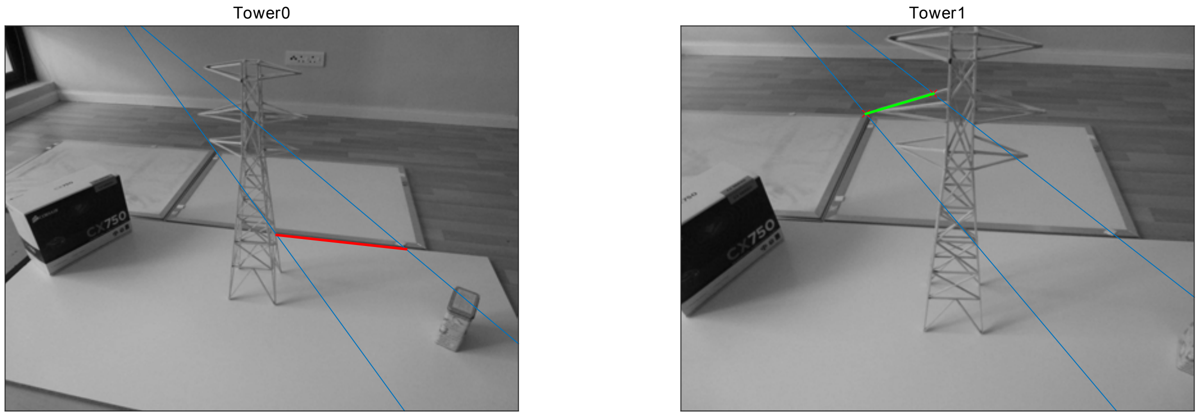


FIGURE 5.9: LSD line (left image) with another valid candidate detected (right image).

Figure 5.9 shows an example of a false correspondence, but yet the segment was still a valid candidate that contributed to the 3D hypotheses set. This procedure was repeated for all line segments detected in I_j where $j = [0, ..., 21]$. If a hypothesis line segment in 2D had endpoints that did not intersect with the epipolar lines, the hypothesis was elongated. Hypotheses in 3D were line segments triangulated from endpoints that intersected epipolar lines. In total, there were 112841 hypotheses (lines) in 3D. This set, S, had to be scored and clustered to form a final reconstruction.

Scoring

The scoring process involved using 22 image-neighbourhoods. Each neighbourhood had members that obeyed two criteria; a valid member was a camera centre not more that 300 mm away, and whose principal axis was, at most, 50° from the image that formed the neighbourhood. The selection for this criteria was based on the distance traversed around the tower and observing the reconstructed, scaled cameras in Figure 5.6. In addition, looking at the images that met this criteria, it was noted that there was enough overlap in terms of the target being viewed. Figure 5.10 shows an example neighbourhood (camera 11).

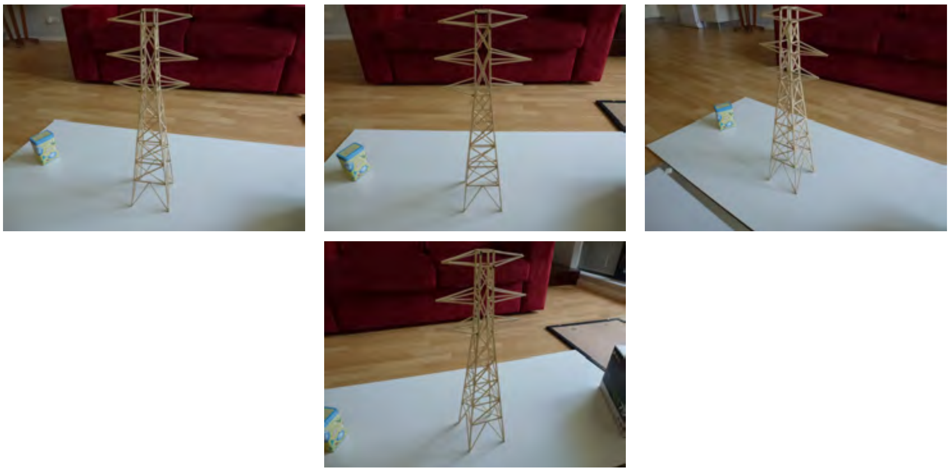


FIGURE 5.10: Neigbourhood 11 with 4 images that scored all hypotheses.

The iterative scoring process coded is provided as pseudo-code in Algorithm 1. The design of the scoring mechanism was not built for speed as this tower reconstruction mechanism is acceptable as an offline process. The slowest part of the line reconstruction process was the $O(n^3)$ scoring procedure. All 112841 hypotheses were scored in 20 hours on an i7 computer with 8GB of RAM.

Algorithm 1 Iterative scoring

procedure SCORING($\{S\}$)
  for `<Each Neighbourhood` N`>` **do**
    for `<Each Neighbour` I `>` **do**
      $pts2D \leftarrow 3D - 2D$ projections of $\{S\}$ ▷ Camera matrices used
      for `<Each hypothesis` L_m `>` **do**
        $VI \leftarrow$ image gradient of I
        $score(L_m) \leftarrow score(L_m) +$ update
      end for
    end for
  end for
  return $score$ ▷ Vector of cumulative scores
end procedure

To further explain the scoring process, the 3D hypotheses were projected into each member of the 22 image neighbourhoods. The scoring formula, Equation 4.1, is shown again for convenience,

$$s(L_m) = \frac{1}{|N(I_i)|} \sum_{I \in N(I_i)} \sum_{\mathbf{x} \in M(I)} \frac{||VI(\mathbf{x})||}{|M(I)|} e^{-\left(\frac{\lambda . dist(\mathbf{x}, L_m)}{2 . dist_{max}(L_m)}\right)^2}.$$

The score for a particular hypothesis L_m was based on the image gradient of a neighbour onto which this segment was reprojected. The score was normalized using the size of the neighbourhood in question. The number of discrete measurement points, perpendicular to the image of L_m, was set to 11 and spanned the length of L_m at 5 pixel intervals. These measurement points were locations at which the underlying image gradient was used to score a reprojected hypothesis. In terms of representing an image gradient, the Sobel operator was avoided due to its noise sensitivity. More noticeably, a Sobel operator could not always provide edges that conformed to the real target. This is depicted in Figure 5.11.

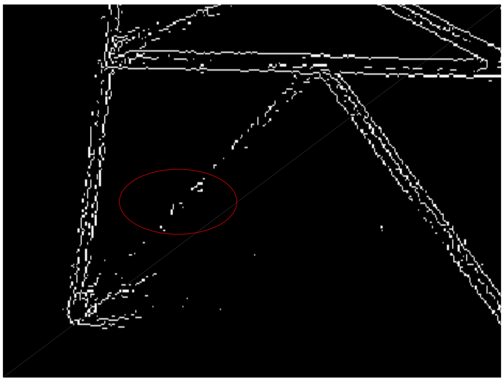
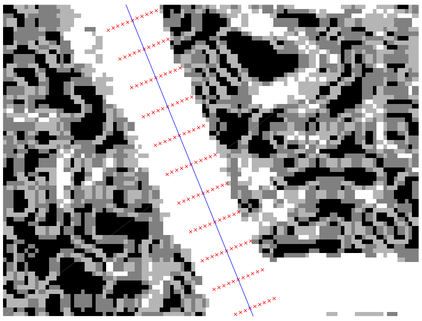


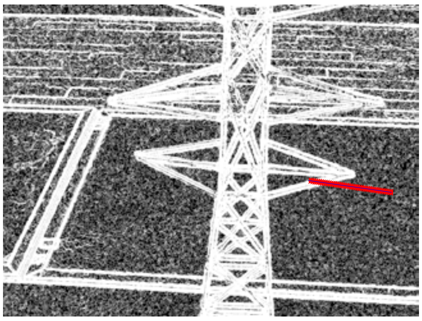
FIGURE 5.11: Sobel edge detector on one of the images.

Figure 5.11 shows line segments that are discontinuous and irregular. As a consequence, MATLAB's *imagegradient* function (central difference method) was used to obtain the image gradients for all image neighbours. The image gradient had to maintain the structure of the tower. These would influence the scores of hypotheses and false matches had to be discriminated against properly. Examples of a good and noisy candidate, in terms of its respective reprojection back into the image space, is shown in Figure 5.12.

The noisy hypothesis in Figure 5.12B arose from the previous phase, whereby a false match could not be discriminated against. Since all 22 neighbourhoods were used, there was sufficient overlap to reinforce a discrepancy between a valid hypothesis and a noisy candidate. To illustrate, Figure 5.13 is provided.



(A) An example of a valid hypothesis.



(B) An invalid match.

FIGURE 5.12: An example of a good (A) and noisy (B) candidate in the image space.

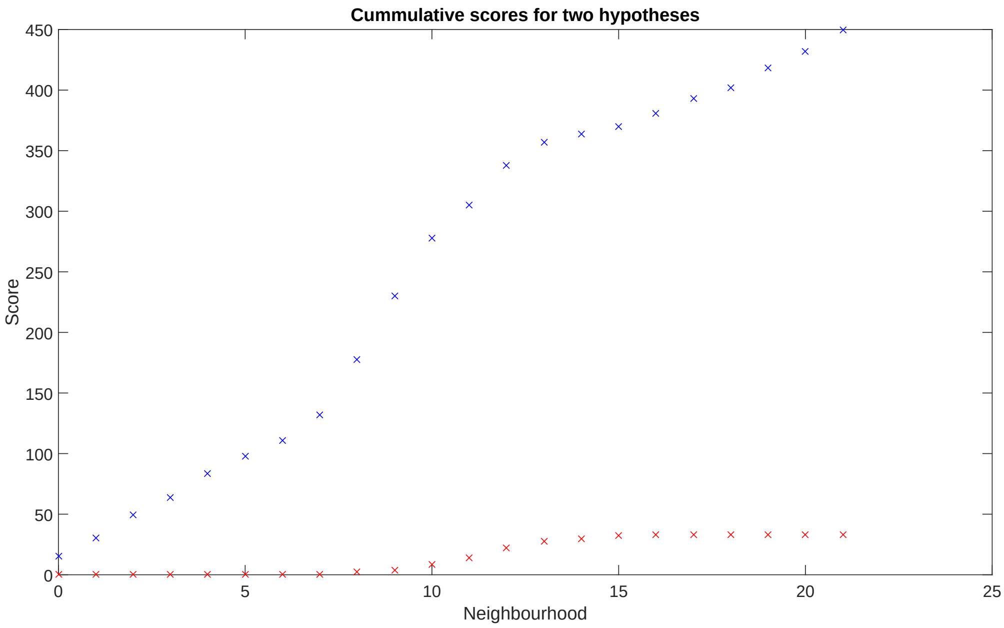


FIGURE 5.13: Difference between a good hypothesis and noisy candidate.

However, despite this expected discrepancy between a valid and invalid match, the graph in Figure 5.13 required inspection as to where the scores for noisy candidates came from. A notable cause was due to a partial overlap between a line from the image space and the false candidate in question (Figure 5.12B). These overlaps cannot be prevented from occurring in several reprojections and these too would accumulate over such occurrences. Figure 5.12B shows that the image gradient also has background noise that would contribute to the overall score of false hypotheses. These observations would have had compounding effects on the scores of invalid matches. To provide an overview, all scores for all hypotheses (with all image neighbourhoods considered) were sorted and the following log graph is provided.

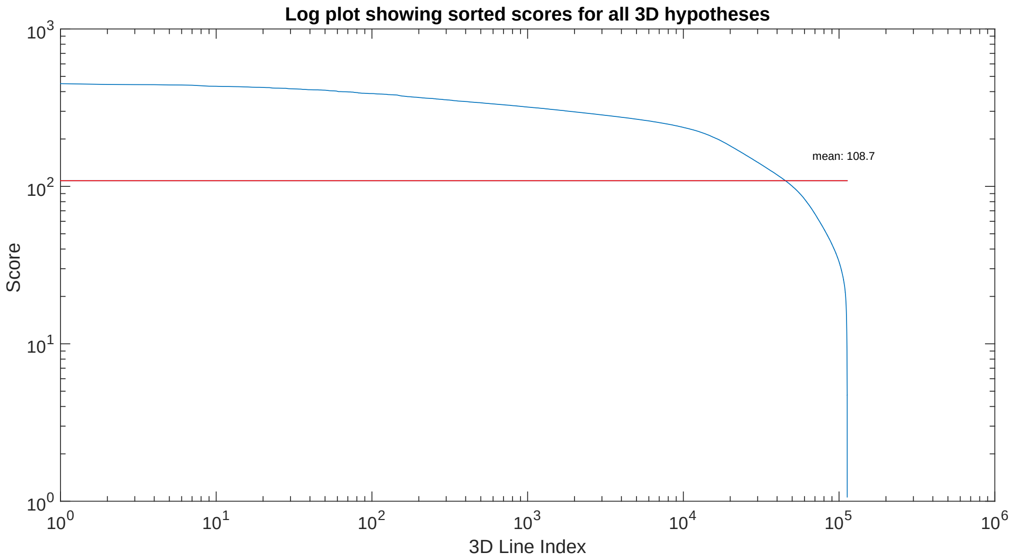


FIGURE 5.14: Log plot showing scores for 3D candidates.

Figure 5.14 indicates the 3D line index (for the sorted list) on the horizontal axis. The average score was 108.7, and in addition to clustering parameters this knowledge was used during the reconstruction step. More specifically, the clustering process only created cylindrical volumes for lines that had a score of at least 108. (This does not, however, stop a low scoring line from being accepted into an existing cluster volume).

Clustering

The two clustering parameters used were $r = 3$ mm and $H_{min} = 3$. These remain fixed throughout the clustering process. The radius was based on the thickness of the balsa wood used in the construction of the model tower. The second threshold was based on the average size of image neighbourhoods. Unlike [16], there was no lengthening of line segments during the clustering phase. This was because a decision was made to have a stricter clustering mechanism. The result of this strict clustering process is shown in Figure 5.15.

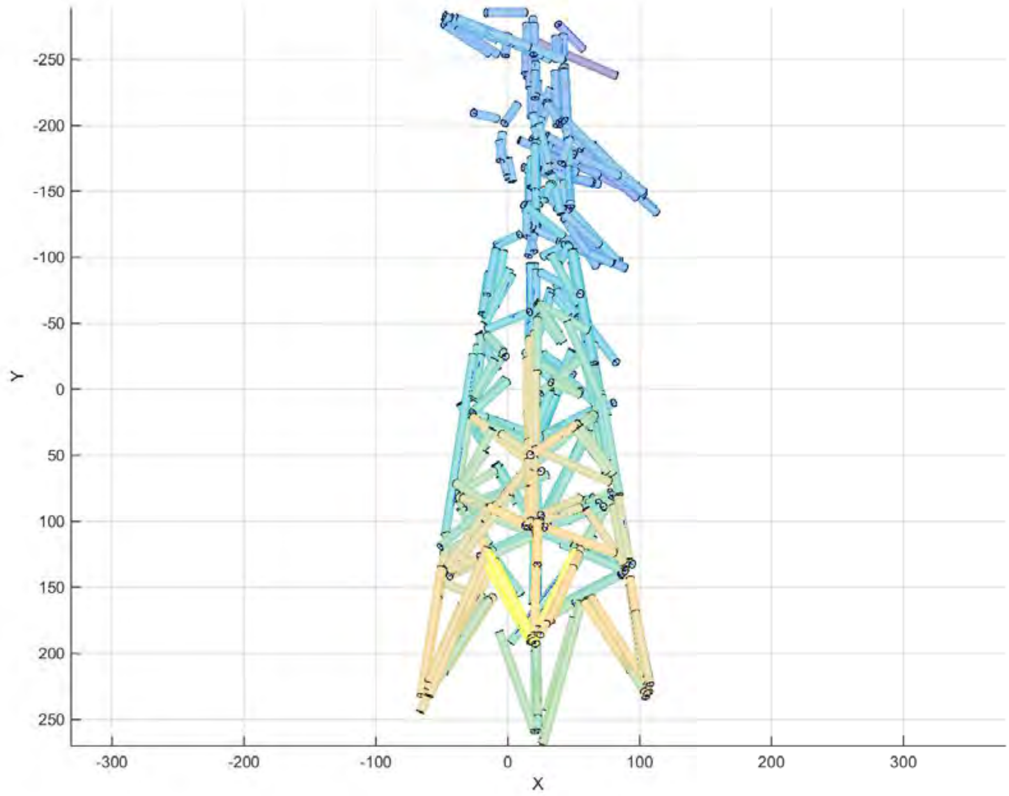
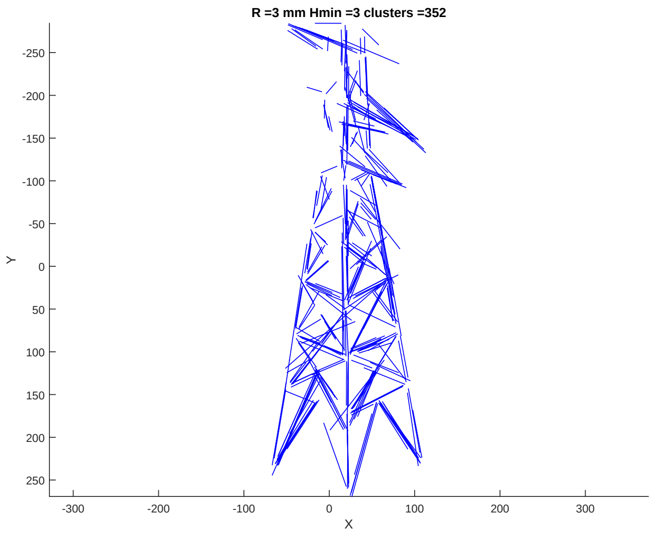


Figure 5.15: All clusters for model tower.

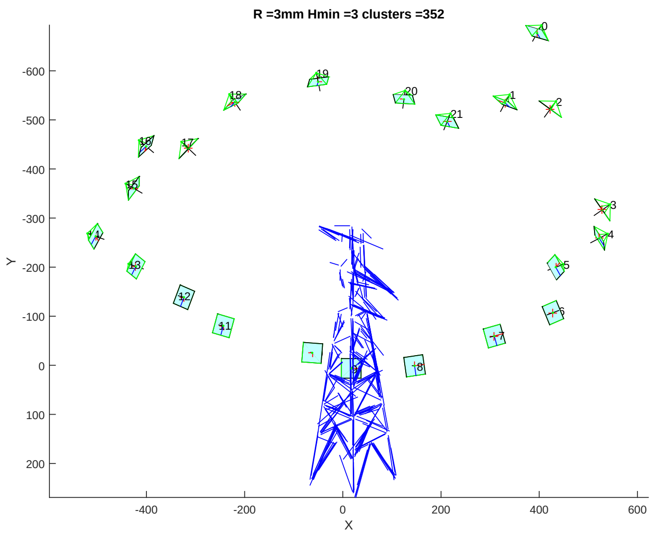
The general shape of the tower can be seen but there are missing segments at the top. This is a consequence of having a stricter clustering mechanism. It was also realised that the clustering mechanism did not account for repeated line segments (multiple edges representing the same line segment on the tower) since each clustered line did not acknowledge the presence of another cluster volume that might be in close proximity.

Reconstruction

Following the clustering process, the tower reconstruction obtained is shown in Figure 5.16A. The tower is correct to scale and cameras are shown together with the model in Figure 5.16B.



(A) Final Tower Reconstruction.



(B) Normalized, scaled cameras added.

FIGURE 5.16: Reconstructed cameras and transmission tower.

There are missing line segments in the model depicted in Figure 5.16A. This is a direct result of only considering lines that had a score of at least 108. During this clustering process, about 50 000 lines (half the data set) were rejected (see the Figure 5.14). The objective was to reduce the number of spurious lines and this required a trade off whereby the number of reconstructed segments towards a full tower was reduced. Noisy lines were a result of image gradients having background noise and false candidates accumulating scores based on background noise. More noticeably, there are repeated line segments depicting a tower line segment. This is because the clustering process is not designed to be aware of multiple clusters that may be parallel and close to each other.

The parts of the tower that have been reconstructed are correct to scale. Figure 5.16B shows the final reconstruction along with scaled cameras. This conforms to the real setup. To verify scale, reprojecting some of the final lines back into the image space was done. In Figure 5.17, the line index 3 is incomplete but is 113 mm. The length of the same segment is 120 mm in reality. In addition, the measured height of the tower is 590 mm, whereas the reconstructed version represents a tower with height 578 mm.

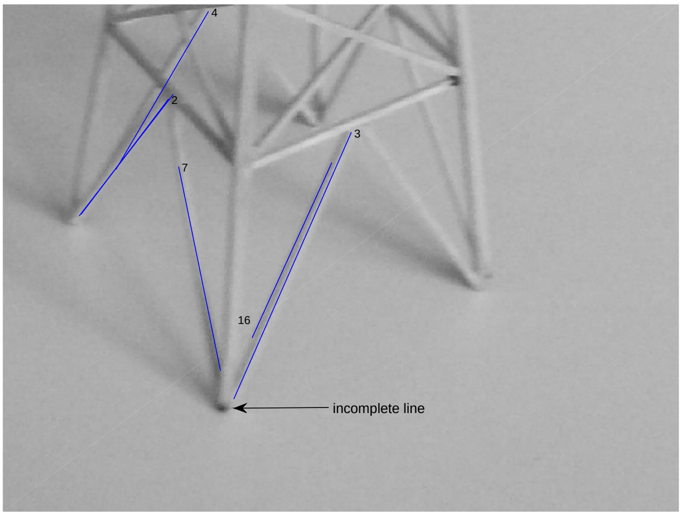


Figure 5.17: Reconstructed leg segment.

The shortfall in expected dimensions is a combined effect of the scoring and clustering procedures followed. Missing line segments in 3D can also be related to a lack of representation in 2D (missed detections by the LSD algorithm due to contrast). However, the exercise overall showed that this is a feasible pipeline where image data, a combination of image-space algorithms and camera geometry can be coordinated in a manner that reconstructs a tower. Even though this model is not complete, a significant portion of the tower is represented.

5.2.1 TU GRAZ dataset

The same line reconstruction pipeline that was coded for the balsa tower was applied to data provided by [16]. This exercise was to verify the code developed in MATLAB. The dataset consisted of 106 images and their respective camera models.

5.2.2 Real tower reconstruction

Using the code developed in this work, the reconstruction algorithm took a week to run on an i7 computer. Clustering parameters (r and H_{min}) were set to values prescribed in [16] to try and meet the expected results. Figure 5.18 shows some of the image data that were captured in [16]. It is noted immediately that the tower is the desired subject in each of the images.

FIGURE 5.18: Tower images obtained from [15] and used in [16].

The images in Figure 5.18 were taken in a way that prevented other objects in the background from competing with the tower as the main focus. Background clutter was inevitable and there was glare from the sun. The contrast of the tower with the blue sky was advantageous for the LSD algorithm. After hypotheses generation, scoring and clustering was done.

Figure 5.19 shows the reconstructed model (units in metres). For cylinder formation, lines in 3D were lengthened by 10% while clustering parameters were $r = 1$ cm and $H_{min} = 3$. This was to adhere to the values selected [16]. In the final reconstruction, unwanted lines were present but the underlying 3D tower was still distinct. Every image I_i had an opportunity to create 3D hypotheses after triangulating lines with image I_{i+5}. The set of 3D hypotheses, S, for this experiment, contained 524700 lines. The final reconstruction had only 983 model line segment. Figure 5.19 shows this reconstruction.

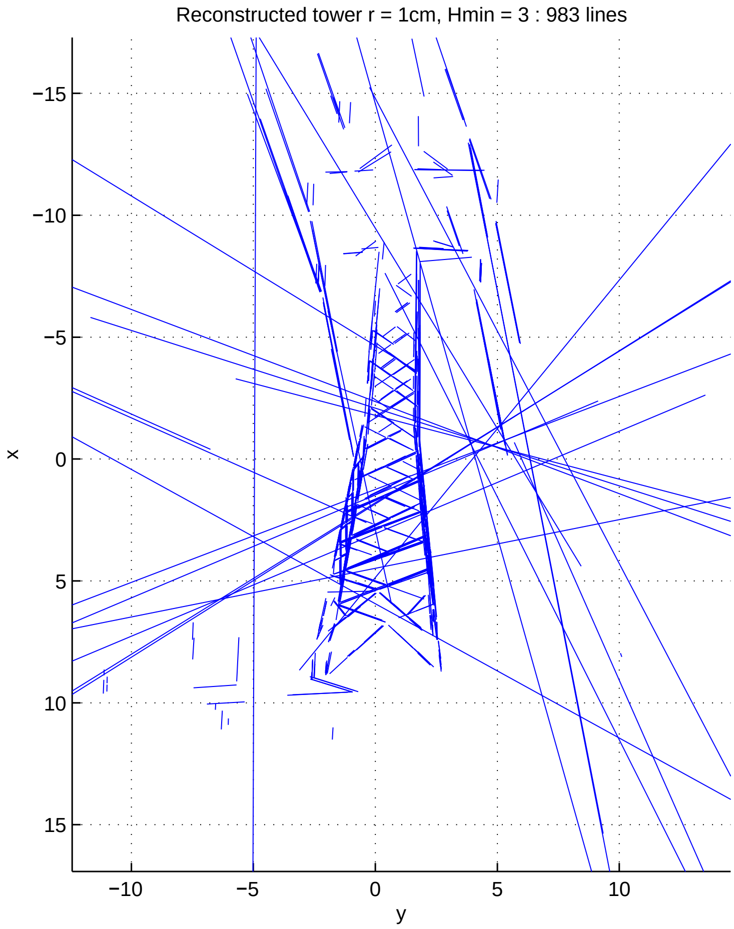
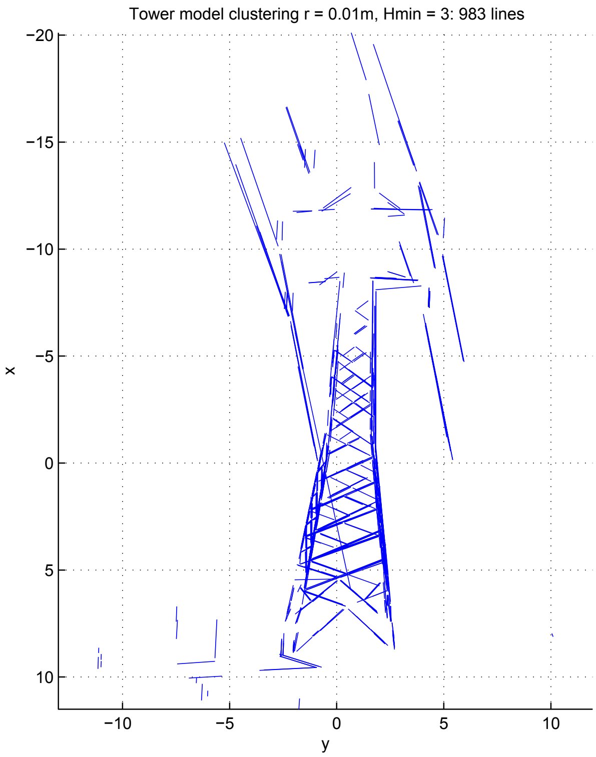
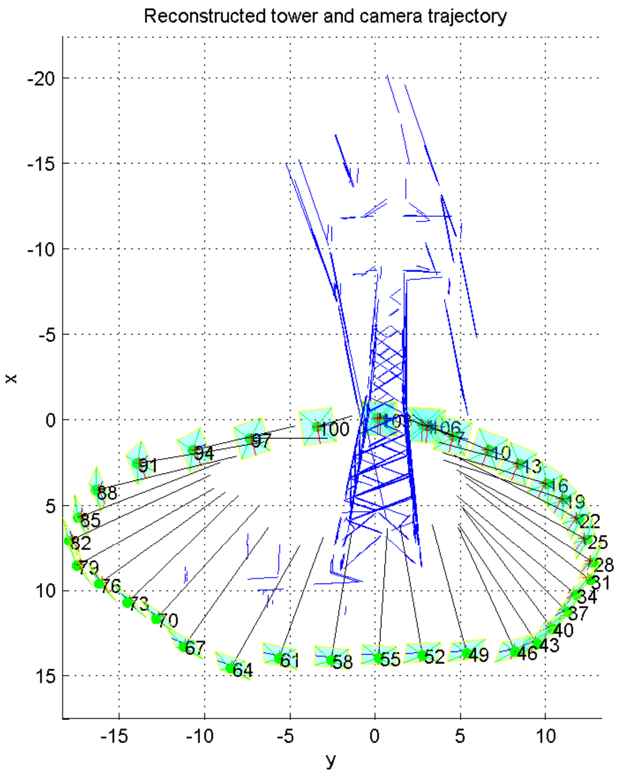


FIGURE 5.19: A result resembling the tower obtained in [16] — but with spurious lines.

The height of the tower in Figure 5.19 is approximately 20 m. This was a realistic value. Model lines exceeding 30 m were removed from the reconstruction in Figure 5.19 and fewer spurious lines obstructed the desired tower. A stricter clustering step could have prevented the need to manually remove noisy lines. Figures 5.20A and 5.20B illustrates this. In addition to the reconstructed tower, Figure 5.20B also shows the camera trajectory pursued by [16].



(A) Reconstructed model excluding lines exceeding 30 m in length.

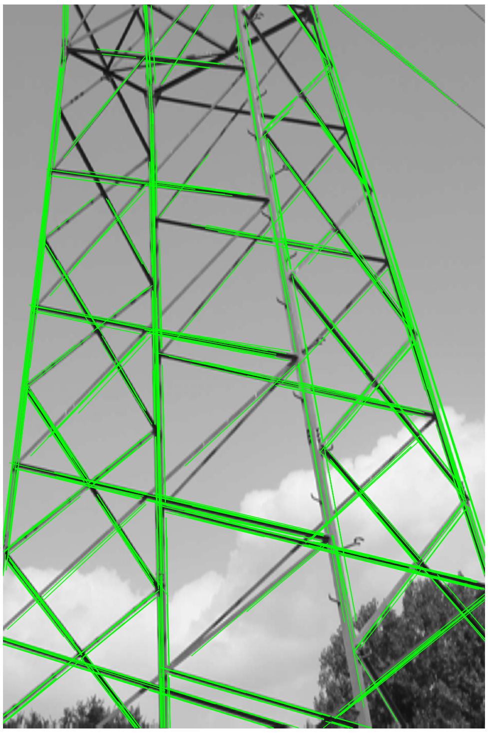


(B) The forward-principal axis of every camera faces the tower.

FIGURE 5.20: Reconstruction of a real tower.

The cameras that captured the tower along its circular trajectory are shown in Figure 5.20B. A reprojection of the final 3D model (Figure 5.20A) onto one of the images shows how well the model lines project onto the image. Figure 5.21A is provided to show the reprojection, while Figure 5.21B provides a close up view.

(A) Reprojection of tower model onto one of the images.



(B) A close up of the reprojection.

Figure 5.21: Reprojection of model into an image.

Figure 5.21B is a close up of the reprojected model and it can be seen that some lines of the tower were not reconstructed. Multiple edges also exist. This arose from, as observed from previous experiments, fixed clustering parameters that remain constant throughout the process. The clustering mechanism may be tolerant compared to the balsa tower reconstruction, but edges of a tower segment are represented with multiple 3D model lines. Despite these observations, the structure of the tower itself is captured.

5.3 Overview of reconstruction pipeline

The reconstruction pipeline is feasible for building a tower model correct to scale. The dependency chain is important and starts with establishing correct camera extrinsics, scale, 2D line representations and using valid image gradients. For the line reconstruction pipeline, these observations are made.

Hypotheses generation

The LSD algorithm is susceptible to missing lines. These arise from lighting issues (glare or contrast issues) and could cause the final reconstruction to have missing line segments. It is not simply overcome by relying on more images to participate in the reconstruction process. Detection of 2D features is not decoupled from lighting and occlusion conditions as indicated in the work acknowledged in Chapter 2.

Scoring

The scoring process is an $O(n^3)$ operation that is feasible for offline circumstances. The nature of image gradients is also crucial. Image gradients of lines need to be regular and continuous to facilitate a fair scoring process. However, this must not be compensated for to the extent that background noise allows for a significant increase of false scores. Image neighbourhoods are based on parameters set by a human thereby indicating that there is a level of human dependency in the scoring process as well.

Clustering

The clustering process relies on parameters that are not flexible across all hypotheses. Cylindrical volumes themselves are not clustered if they are parallel to each other and this leads to multiple edges representing a real line segment. Hypotheses accepted into the clustering volume are also not required to be parallel to the principal axis.

Chapter 6

Conclusions and future work

The focus of this work was to propose a 3D model of a transmission tower using camera geometry and image data. Furthermore, it was shown how features that are directly related to a wiry target object (lines in this case) can be exploited in a reconstruction problem. This was done by considering the tower as a 'wiry' object. Popular algorithms like SIFT are limited in their ability to represent corner points on objects that lack texture (a tower).

State-of-art work is heading in the right direction by detecting primitive shapes that are applicable for the power industry. However, for 3D vision, specialised equipment like laser scanners are being used [38], thus bypassing knowledge about feature detection and camera geometry. The algorithm coded in this work can achieve a scaled reconstruction of a tower while the geometry of the target is preserved. Using images and exploiting multi-view geometry to generate such models makes the wiry model reconstruction worthy of research. The model is compact and does not require specialised hardware other than a camera. However, for robot navigation and power line inspection, a more refined tower needs to be reconstructed. This translates to addressing problems faced with the reconstruction pipeline itself. The justification for the algorithms chosen to form this tower reconstruction pipeline is suitable, but there are considerations to be made. These are issues that are well known in all image processing problems.

Lighting issues and contrast effects have an impact on hypothesis generation. The LSD algorithm may not detect these segments. For the balsa tower reconstruction, poor contrast could not be compensated with more images, as initially expected. The reconstruction of a real tower [16] involved using images in which glare was present. As a result, some tower segments were not reconstructed. The tower reconstruction presented is not free of lighting effects and is susceptible to missing data as described in section 2.1.

The edge detection techniques that score line segments are crucial in discriminating against false candidates. The image gradient used for the reconstruction of the balsa tower is still susceptible to background noise. Transferring this problem to the clustering process (using a score average) does not solve the problem of false hypotheses accumulating a score.

The clustering step of the reconstruction pipeline relies on a user to define parameters that relate to the real object. In addition, these are thresholds that remain fixed throughout the clustering step. The end result is that the proposed reconstruction has repeated line segments.

Challenges of the current reconstruction pipeline

This research was an exercise that applied a sequence of algorithms to facilitate tower reconstruction. In retrospect, expectations were constructed such that if an algorithm could not achieve its object to the fullest extent (for instance, LSD failing to detect all 2D line segments), these could be passed to another procedure that might provide some compensation. This should not be the case. As in [27], the resulting pipeline is a series of dependent algorithms. It may be beneficial to have contingencies in some of the reconstruction steps.

For 2D detection, if LSD is the main algorithm responsible for detecting lines, it can consult the output of a Hough transform to see if any lines were missed. The 2D representation of a tower must be exhausted as far as possible in order to proceed with a 3D representation. Another challenge includes having the clustering process produce line segments that are unique in orientation and position. Multiple line segments should not have to represent a reconstructed line segment. Having greater 2D representation and fewer unnecessary (reconstructed) line segments would reinforce the current approach and produce a more complete tower representation.

Future work

Firstly, improving the computational effort for this type of pipeline will benefit all other improvements. Quicker results that scale well with the number of images will speed up development. As for the individual steps in the reconstruction pipeline, a suggested approach is to have a two-step 2D line detection algorithm that does not rely on a single algorithm (LSD) for line detection. Secondly, a better clustering mechanism that adapts to the score of the line being investigated would lead to a better tower reconstruction. This clustering step should be aware of other line segments that are in close proximity and perform additional clustering if necessary.

Bibliography

[1] Dana H Ballard. Generalizing the Hough transform to detect arbitrary shapes. *Pattern recognition*, 13(2):111–122, 1981.

[2] Daniel F. Dementhon and Larry S Davis. Model-based object pose in 25 lines of code. *International journal of computer vision*, 15(1-2):123–141, 1995.

[3] Agnès Desolneux, Lionel Moisan, and Jean-Michel. Morel. Edge detection by Helmholtz principle. *Journal of Mathematical Imaging and Vision*, 14(3):271–284, 2001.

[4] Agnes Desolneux, Lionel Moisan, and J-M Morel. *From gestalt theory to image analysis: a probabilistic approach*, volume 34. Springer Science & Business Media, 2007.

[5] Eskom. Electricity: Costs and Benefits, 2014. URL http://www.eskom.co.za/AboutElectricity/FactsFigures/Pages/Facts_Figures.aspx.

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

[7] David A. Forsyth and Jean Ponce. A modern approach. *Computer Vision: A Modern Approach*, pages 88–101, 2003.

[8] Siyao Fu, Weiming Li, Yunchu Zhang, Zize Liang, Zengguang Hou, Min Tan, Wenbo Ye, Bo Lian, and Qi Zuo. Structure-constrained obstacles recognition for power transmission line inspection robot. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 3363–3368. IEEE, 2006.

[9] Siyao Fu, Zize Liang, Zengguang Hou, and Min Tan. Vision based navigation for power transmission line inspection robot. In *2008 7th IEEE International Conference on Cognitive Informatics*, pages 411–417. IEEE, August 2008. ISBN 978-1-4244-2538-9. doi: 10.1109/COGINF.2008.4639195. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4639195.

[10] Rafael Grompone von Gioi, Jean-michel Morel, and Gregory Randall. LSD : a Line Segment Detector. *Image Processing On Line*, 2:35–55, 2012. doi: 10.5201/ipol.2012.gjmr-lsd.

[11] Ian Golightly and Dewi Jones. Corner detection and matching for visual tracking during power line inspection. *Image and Vision Computing*, 21(9):827–840, 2003.

[12] Rafael C. Gonzales and Richard E. Woods. *Digital Image Processing, 2nd Edition*. Prentice Hall, 2002.

[13] Robert M. Haralick, Stanley R. Sternberg, and Xinhua Zhuang. Image analysis using mathematical morphology. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 9(4):532–550, 1987.

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

[15] Manuel Hofer, Andreas Wendel, and Horst Bischof. personal communication, September .

[16] Manuel Hofer, Andreas Wendel, and Horst Bischof. Line-based 3d reconstruction of wiry objects. In *18th Computer Vision Winter Workshop*, pages 78–85, 2013.

[17] Acroname Inc. HOKUYO UTM-30LX, 2014. URL http://www.acroname.com/products/R314-HOKUYO-LASER4.html.

[18] Powerline Systems Inc. PLS-CADD, 2014. URL http://www.powline.com/products/pls_cadd.html.

[19] Arnold Irschara, Viktor Kaufmann, Manfred Klopschitz, Horst Bischof, and Franz Leberl. Towards fully automatic photogrammetric reconstruction using digital images taken from uavs, 2010. URL http://aerial.icg.tugraz.at/papers/uav_reconstruction_isprs2010.pdf.

[20] Mohammad R. Jahanshahi, Sami F. Masri, Curtis W. Padgett, and Gaurav S. Sukhatme. An innovative methodology for detection and quantification of cracks through incorporation of depth perception. *Machine Vision and Applications*, 24(2):227–241, December 2011. ISSN 0932-8092. doi: 10.1007/s00138-011-0394-0. URL http://link.springer.com/10.1007/s00138-011-0394-0.

[21] Jaka Katrašnik, Franjo Pernuš, and Boštjan Likar. A climbing-flying robot for power line inspection. In *Proceedings of the IEEE Conference on robotics, Automation and Mechatronics*, pages 95–110, 2008.

[22] Laurent Kneip, Davide Scaramuzza, and Roland Siegwart. A novel parametrization of the perspective-three-point problem for a direct computation of absolute camera position and orientation. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 2969–2976. IEEE, 2011.

[23] Wai Ho Li, Arman Tajbakhsh, Carl Rathbone, and Yogendra Vashishtha. Image processing to automate condition assessment of overhead line components. In *Applied Robotics for the Power Industry (CARPI), 2010 1st International Conference on*, pages 1–6. IEEE, 2010.

[24] Nicolas Limare and Jean-Michel Morel. The IPOL initiative: Publishing and testing algorithms on line for reproducible research in image processing. *Procedia Computer Science*, 4:716–725, 2011.

[25] H. Christopher Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. *Readings in Computer Vision: Issues, Problems, Principles, and Paradigms, MA Fischler and O. Firschein, eds*, pages 61–62, 1987.

[26] T. Lorimer. personal communication, August .

[27] Trevor Lorimer. Masters dissertation: The design and construction of a robotic platform for power line inspection. Master's thesis, University of KwaZulu-Natal, South Africa, 2011.

[28] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004. doi: 10.1023/B:VISI.0000029664.99615.94.

[29] Raman Maini and Himanshu Aggarwal. Study and comparison of various image edge detection techniques. *International Journal of Image Processing (IJIP)*, 3(1):1–11, 2009.

[30] MATLAB. *version 8.4 (R2014b)*. The MathWorks Inc., Natick, Massachusetts, 2014.

[31] Rajiv Mehrotra, Kameswara Rao Namuduri, and Nagarajan Ranganathan. Gabor filter-based edge detection. *Pattern Recognition*, 25(12):1479–1494, 1992.

[32] Serge Montambault and Nicolas Pouliot. The HQ LineROVer: contributing to innovation in transmission line maintenance. In *Transmission and Distribution Construction, Operation and Live-Line Maintenance, 2003. 2003 IEEE ESMO. 2003 IEEE 10th International Conference on*, pages 33–40. IEEE, 2003.

[33] Pierre Moulon, Pascal Monasse, and Renaud Marlet. Adaptive structure from motion with a contrario model estimation. In *Computer Vision–ACCV 2012*, pages 257–270. Springer, 2012.

[34] OpenCV. OpenCV library, 2014. URL http://opencv.org/.

[35] openMVG. openMVG, 2015. URL https://openmvg.readthedocs.org/en/latest/openMVG/openMVG/.

[36] Panasonic. Dmc-zs10, 2015. URL http://shop.panasonic.com/support-only/DMC-ZS10S.htm.

[37] James F. Peters, Sheela Ramanna, and Marcin S. Szczuka. Towards a line-crawling robot obstacle classification system: a rough set approach. In *Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing*, pages 303–307. Springer, 2003.

[38] Nicolas Pouliot, P. Richard, and Serge Montambault. LineScout power line robot: Characterization of a UTM-30LX LIDAR system for obstacle detection. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 4327–4334, 2012.

[39] P. H. Pretorius. On particular aspects related to 400 kV Transmission. Technical memo, July 2012. URL http://www.eskom.co.za/OurCompany/SustainableDevelopment/EnvironmentalImpactAssessments/1600/Documents/Tech_Memo_-_400_kV_Lines_-_V0_-_22_Jul_2012.pdf.

[40] Long Quan and Zhongdan Lan. Linear n-point camera pose determination. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 21(8):774–780, 1999.

[41] Pierre-Luc Richard, Nicolas Pouliot, and Serge Montambault. Introduction of a LIDAR-based obstacle detection system on the LineScout power line robot. In *Advanced Intelligent Mechatronics (AIM), 2014 IEEE/ASME International Conference on*, pages 1734–1740. IEEE, 2014.

[42] Noah Snavely, Steven Seitz, and Richard Szelisk. Photo Tourism: Exploring image collections in 3D, 2006. URL http://www.cs.cornell.edu/~snavely/bundler/#S4http://www.cs.cornell.edu/~snavely/bundler/#S4.

[43] African Consulting Surveyors. Transmission Tower 3D scan, 2014. URL http://www.africansurveyors.com/transmission-tower-3d-scan-and-model.html.

[44] Uvirco Technologies. CoroCAM, 2014. URL http://www.uvirco.com/overview.html.

[45] Jittichat Tilawat, N. Theera-Umpon, and S. Auephanwiriyakul. Automatic detection of electricity pylons in aerial video sequences. In *Electronics and Information Engineering (ICEIE), 2010 International Conference On*, pages V1–342. IEEE, 2010.

[46] Kristopher Toussaint, Nicolas Poliout, and Serge Montambault. Transmission line maintenance robots capable of crossing obstacles: state of the art review and challenges ahead. *Journal of Field Robotics*, 26(5):477–499, 2009. doi: 10.1002/rob.20295. URL http://onlinelibrary.wiley.com/doi/10.1002/rob.20295/abstract.

[47] Andreas Wendel, Arnold Irschara, and Horst Bischof. Automatic alignment of 3D reconstructions using a digital surface model. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2011 IEEE Computer Society Conference on*, pages 29–36. IEEE, 2011.

[48] Juyang Weng, Paul Cohen, and Marc Herniou. Camera calibration with distortion models and accuracy evaluation. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (10):965–980, 1992.

[49] CC. Whitworth, AWG. Duller, DI. Jones, and GK. Earp. Aerial video inspection of overhead power lines. *Power Engineering Journal*, 15(1):25–32, 2001.

[50] F.Y. Zhou, J.D. Wang, Y.B. Li, J. Wang, and H.R. Xiao. Control of an inspection robot for 110kv power transmission lines based on expert system design methods. In *Control Applications, 2005. CCA 2005. Proceedings of 2005 IEEE Conference on*, pages 1563–1568. IEEE, 2005.

[51] Djemel Ziou. Line detection using an optimal IIR filter. *Pattern Recognition*, 24(6):465–478, 1991.

[52] Qi Zuo, Zhi Xie, Zijian Guo, and Dehui Sun. The obstacle recognition approach for a power line inspection robot. In *Mechatronics and Automation, 2009. ICMA 2009. International Conference on*, pages 1757–1761. IEEE, 2009.

Appendix A

Images of model tower

The images obtained with a Panasonic camera [36] are shown in Figure A.1.

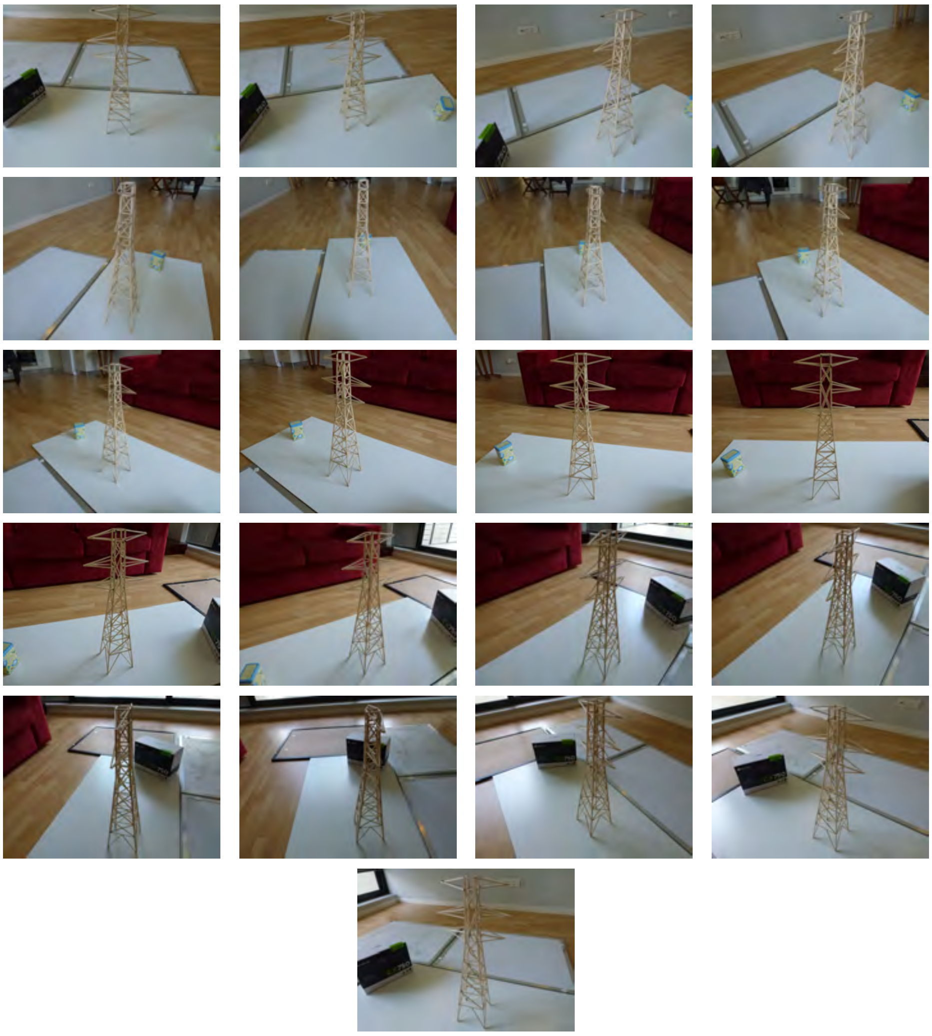


FIGURE A.1: Images 0 to 21 from left to right and top to bottom.