Structure-Aware Monocular Visual-Inertial Navigation of Unmanned Aerial Vehicles (UAVs) in Urban Scenes

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

*(Master of Science in **Electronics and Communication Engineering** by Research*

by

Thatavarthy VVSST Ayyappa Swamy
2019702011
vvsst.ayyappa@research.iiit.ac.in

International Institute of Information Technology
Hyderabad - 500 032, INDIA
November 2022

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled " **Structure-Aware Monocular Visual-Inertial Navigation of Unmanned Aerial Vehicles (UAVs) in Urban Scenes** " by Thatavarthy VVSST Ayyappa Swamy, has been carried out under my supervision and is not submitted elsewhere for a degree.

_____ _____

Date Adviser: Prof. K. Madhava Krishna

To the Past, Present and Future

Acknowledgments

Thanks to IHub-Data, IIIT Hyderabad for research fellowship.

Abstract

Navigation of Unmanned Aerial Vehicles (UAVs) amongst high-rises in urban environments becomes inevitable due to increasing demand for various applications in Urban Air Mobility (UAM). Moreover, these city scale urban environments are populated with tall buildings and skyscrapers with inherent piecewise planar structures. Leveraging this inherent structural information for navigating a UAV equipped with a single camera and an IMU is the primary focus of this thesis. We make three contributions that leverage these geometric cues to detect and map these planar structures for obstacle avoidance and path planning.

Our first contribution, **Multi-View Planarity Constraints for skyline estimation from UAV images in city scale urban environments**, is a three-stage pipeline that brings together several modules combining a data-driven monocular plane segmentation network and geometric constraints from 3D vision to showcase a quick reconstruction of planar facades within a few views. We evaluate the efficacy of our pipeline with various constraints and errors from multi-view geometry using ablation studies. We then retrieve the skyline of the buildings in synthetic as well as real-world scenes.

In our second contribution, **A new geometric approach for three view line reconstruction and motion estimation in Manhattan Scenes**, we propose a novel method of pose estimation using line features from three views of a Manhattan Scene. We leverage the vanishing point directions to estimate the relative rotations as well as to fix the 3D line direction. In consequence we build a constraints matrix, which has the relative translations and 3D line depth as its null space. We then perform 1-parameter line BA using factor graph based cost function. We compare the efficacy of our method with standard line triangulation in synthetic as well as real-world scenes.

Finally, we propose **UrbanFly: Uncertainty-Aware Planning for Navigation Amongst High-Rises with Monocular Visual-Inertial SLAM Maps**, a sequential convex program based novel trajectory planner tailored for outdoor urban scenes. It uses the sparse point clouds generated by a Monocular Visual Inertial SLAM (VINS) backend to build a cuboid representation of the environment through a data-driven monocular plane segmentation network. Our chosen world model provides faster distance queries than the more common voxel-grid representation. The trajectory optimizer is initialized based on safety estimates on a set of randomly drawn trajectories. We perform extensive simulations in a tightly integrated perception-control loop to validate its efficacy. UrbanFly outperforms competing baselines in metrics such as collision rate, trajectory length, etc., on a high fidelity AirSim simulator augmented with synthetic and real-world dataset scenes.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Recent developments in robotics have enabled the usage of UAVs for various applications like aerial surveying and inspection, package delivery, etc., However, navigation of UAVs in urban scenes pose various challenges for localization, mapping and path planning.

UAVs (Unmanned Aerial Vehicles) are constrained in terms of payload capability, compute power and battery life. Other constraints arise due to the existing sensing methods. LiDARs are heavy and demand power from the battery. Stereo and depth cameras are not suitable for long range. Multi view stereo methods are computationally intensive. In contrast, we use a single camera and an IMU which are lightweight and consume less power compared to other methods.

In addition to the innate challenges mentioned above, navigation of UAVs in outdoor urban scenes populated with tall structures like buildings and skyscrapers pose various problems. Inaccurate GPS data due to multi-path fading, lack of external markers used in indoor scenes, lack of ground plane visibility, texture less areas that lack visual features are some of the challenges for localization. Also, path planning is such environments should avoid failure of SLAM (Simultaneous Localization And Mapping) while considering the uncertainty in reconstructed planar map.

Therefore, in this thesis we aim to implement a pipeline that leverages the inherent planar structure of the buildings for mapping and navigation of a UAV (Unmanned Aerial Vehicle) using monocular RGB images and IMU data. We combine various modules that use deep learning and 3D vision to obtain a cuboidal representation of the skyline suitable for path planning.

Leveraging structural information inherent to the buildings in urban scenes is the core aspect of this thesis. Firstly, we propose a novel pipeline for quick dense reconstruction of building facades within few views. We then use this information in the form of vanishing points to estimation the motion of three views of a Manhattan scene.

1.1 Contributions

Key contributions of this thesis can be summarized as follows:

- a pipeline with several modules combining deep learning and 3D vision to showcase a quick reconstruction of the skyline within a few views
- a novel way of assimilating different geometric constraints from multiple views for simultaneous initialization of multiple planar structures
- a study of the efficacy of various structural/geometric constraints, nascent to the 3D vision literature, on initialization, and bundle adjustment
- a new geometric approach for joint estimation of three view structure and motion from line features in Manhattan scenes
- 1-parameter based cost function that allows the use of hyper edge in the factor graph optimization to increase robustness to line feature correspondence outliers
- a pipeline to demonstrate robust and collision-free autonomous flight with a single camera system
- We show advantages of our chosen cuboid representation of the urban-rise environment as compared to the more common voxel-grid representation.

1.2 Thesis Organisation

Chapter 2 presents the works related to this thesis. Multi-view planarity constraint based dense reconstruction of facades is described in Chapter 3. We then present a line feature based motion estimation method from three views of a Manhattan scene in Chapter 4. In Chapter 5, a novel monocular visual-inertial end-to-end navigation framework is presented. Finally, the thesis concludes in Chapter 6 mentioning the possibilities for future work.

Chapter 2

Background

In the chapter we present the works related to the mapping, pose estimation and navigation using monocular visual-inertial sensors in urban scenes.

2.1 Mapping of urban scenes

Reconstructing 3D geometry of an urban scene within a few views from UAV images is not a well-studied problem in Computer Vision literature.

Some methods [54] use deep neural network architectures to predict depth and motion from videos simultaneously. SLAM methods like [29], PL-SLAM[12] only focus on landmarks for localization and mapping. These landmarks are usually sparse and not very useful for estimating the skylines in outdoor environments. They also require a good set of features to track and initialize their system.

While there are many methods for detecting and recovering building structures from aerial and satellite images, approaches that reconstruct using low altitude images are difficult to find in the literature.

For facade detection and segmentation from images, [3] have used LiDAR data.

There are a few learning based frameworks like PlaneNet [32], PlaneRecover [50] and PlaneRCNN [30] to detect planes from 2D images. Both PlaneNet and PlaneRecover have a limitation on the number of planes detected 10 and 5, respectively. We use PlaneRCNN for plane segmentation, as it uses a detection network to overcome this limitation.

Most of the existing methods estimate the layout of an urban scene from a single 2D image. These are not directly useful to build a meaningful local map useful for path planning and navigation. Some of the single view based approaches that work in outdoor urban scenes are as follows. [55] have proposed a method to recover 3D wireframe from single view images. Some other methods like [40] and [41] use vanishing points and lines to lift the 2D features to 3D by imposing geometric constraints.

[47] have proposed the notion of the Manhattan-Frame (MF) model to formalize the surface normals of orthogonal and parallel planar structures in man-made environments. Given a set of surface normals or vanishing points, [22] estimate the MF in near real-time and apply it to estimate multiple MFs. In general, a manhattan scene is described by two mutually perpendicular vanishing points and a

vertical vanishing point. To model more complex urban scenes, [45] have proposed Atlanta World with more than two horizontal vanishing point directions. [28] have used Atlanta World based constraints to improve line based SLAM.

[24] proposed geometric constraints for single-view reconstruction of buildings with a user-guided interface. To avoid the depth ambiguity due to projective single view geometry, they also assume that a reference plane such as the ground or one of the building facades is reconstructed in 3D. This assumption cannot be made when reconstructing the buildings in real-time.

Instead of a user-guided interface, the method described in chapter 3 uses PlaneRCNN to automatically segment the individual planar facades of buildings in the scene. We then utilize the information from nearby views in the form of various planar constraints to avoid the need for a reference plane.

2.2 Pose estimation in urban scenes

Most of the existing methods estimate the layout of an urban scene from a single 2D image. These are not directly useful to build a meaningful local map useful for path planning and navigation. Some of the single view based approaches that work in outdoor urban scenes are as follows. [55] have proposed a method to recover 3D wireframe from single view images. Some other methods like [40] and [41] use vanishing points and lines to lift the 2D features to 3D by imposing geometric constraints.

[47] have proposed the notion of the Manhattan-Frame (MF) model to formalize the surface normals of orthogonal and parallel planar structures in man-made environments. Given a set of surface normals or vanishing points, [22] estimate the MF in near real-time and apply it to estimate multiple MFs. In general, a manhattan scene is described by two mutually perpendicular vanishing points and a vertical vanishing point. To model more complex urban scenes, [45] have proposed Atlanta World with more than two horizontal vanishing point directions. [28] have used Atlanta World based constraints to improve line based SLAM.

Even though one-parameter based 3D line optimization is used in StructPLSLAM, they rely on point features to initialize the camera poses. Additionally, they initialize the 3D line using a stereo camera. In contrast, we estimate the motion simultaneously in the initialization step along with the 3D line using 2D line observations in three views of a monocular camera.

Recently, [52] have proposed an uncertainty based map culling algorithm to filter the outliers of multiview line triangulation. Even though the approach proposed by [20] has been very effective in quick reconstruction 3D lines, they too rely on accurate poses recovered using a standard point based motion estimation.

Standard three view line feature based motion estimation and line triangulation as described in [18] involves multiple steps. Firstly, line features are used to compute an the trifocal tensor which has 27 unknown parameters. The camera poses are then recovered from trifocal tensor by computing the epipoles. The recovered poses are then used for triangulation using recent methods like [52]. Estimated lines from triangulation are then optimized using a line BA.

2.3 Navigation in urban scenes

Monocular SLAM based Planners: Literature dealing with trajectory planning and/or navigation with monocular SLAM is sparse when compared with a much larger volume of literature that deals with planning with SLAM [26], [14]. A large part of planning with SLAM focuses on actively moving to places where the state uncertainty (often measured as the trace of the state co-variance) is low [26], [21]. In contrast, the Monocular SLAM frameworks tend to rely on heuristics [37] due to the difficulty of estimating accurate co-variance in a monocular setting. In [7] photometric co-variance is considered with state co-variance propagated over the tangent manifold. Nonetheless, this technique relies on Gaussian noise models, which we aim to relax in our current work.

None of the above frameworks actively incorporate uncertainty-aware collision avoidance constraints into their approach. In [4] an occupancy grid-based planner is proposed in 2D space that does not include uncertainty models and lacks trajectory sample diversity. As we have seen in [17] lack of diversity can result in collisions under duress of perception noise.

Chance Constrained Optimization: Chance constrained optimization [56] has proven to be a popular paradigm for trajectory planning under uncertainty, wherein the probability of collision avoidance is balanced with trajectory smoothness, arc-length, etc. Most of the current works focus on obtaining tractable reformulations of collision probability. A special case arises when the underlying uncertainty is Gaussian, and the obstacle avoidance constraints are approximated in terms of affine form [13]. In contrast, sampling-based approaches such as [6] can adapt to more general setting in terms of both uncertainty and obstacle geometry.

More recently, the extension of chance-constrained optimization to non-parametric spaces using Hilbert Space Embedding was shown to be efficacious [38]. Our work adopts a similar embedding but differs from the cited work in that we plan over multiple steps instead of just reactive one-step planning. The closest to our work is [17] which considered the uncertainty in the voxel grid representation of the world. In contrast, we adopt a more geometrical description of the obstacles specifically suited for urban high-rise environments. Furthermore, our chosen world model also forms the basis of improvement over [17].

Chapter 3

Multi-View Planarity Constraints for skyline estimation from UAV images in city scale urban environments

Faster navigation of drones, in urban environments, is a challenge as buildings and skyscrapers hinder the long-range capability of on-board cameras. A dense reconstruction of the scene within a few views enables incremental path planning.

Therefore, this chapter aims to propose a three stage pipeline as depicted in Figure 3.1 to reconstruct the skyline of buildings using only 3-5 images of the scene captured by an Unmanned Aerial Vehicle (UAV) or an aerial robot. The three stages are 1) pre-processing of the images, 2) initial estimation of the sparse structure followed by its refinement using a modified bundle adjustment, and 3) retrieval of the skyline of the scene by performing a dense reconstruction.

City scale urban environments are populated by buildings with inherent piecewise planar structures. To leverage these geometric cues, we employ a deep neural architecture, PlaneRCNN proposed by [30], to detect visible planar facades with their segmentation masks in the images.

We then extract the notable features such as line junctions, vanishing points, and vanishing lines from each detected plane mask. Orientation of the plane segments is estimated by computing their normals using their corresponding vanishing lines. The geometric constraints that bind the line junctions to the planes in 3D are deduced from multiple views and are stacked into a single constraints matrix. Solving this matrix's null space gives an initial sparse estimate of the piecewise planar structure.

This structure is then refined using a modified bundle adjustment step to minimize a combination of residual terms, as explained in section 3. The facade masks are then projected onto the refined sparse structure to obtain a dense reconstruction. Skyline of the buildings is then retrieved using the dense reconstruction.

Our contributions are:

- a pipeline with several modules combining deep learning and 3D vision to showcase a quick reconstruction of the skyline within a few views.
- a novel way of assimilating different geometric constraints from multiple views for simultaneous initialization of multiple planar structures.

- a study of the efficacy of various structural/geometric constraints, nascent to the 3D vision literature, on initialization, and bundle adjustment.

The chapter is organized as follows. In section 2, we list the works related to the current approach, and in section 3, the methodology of our pipeline is presented. The results of our experiments are described in section 4, followed by conclusions in section 5.

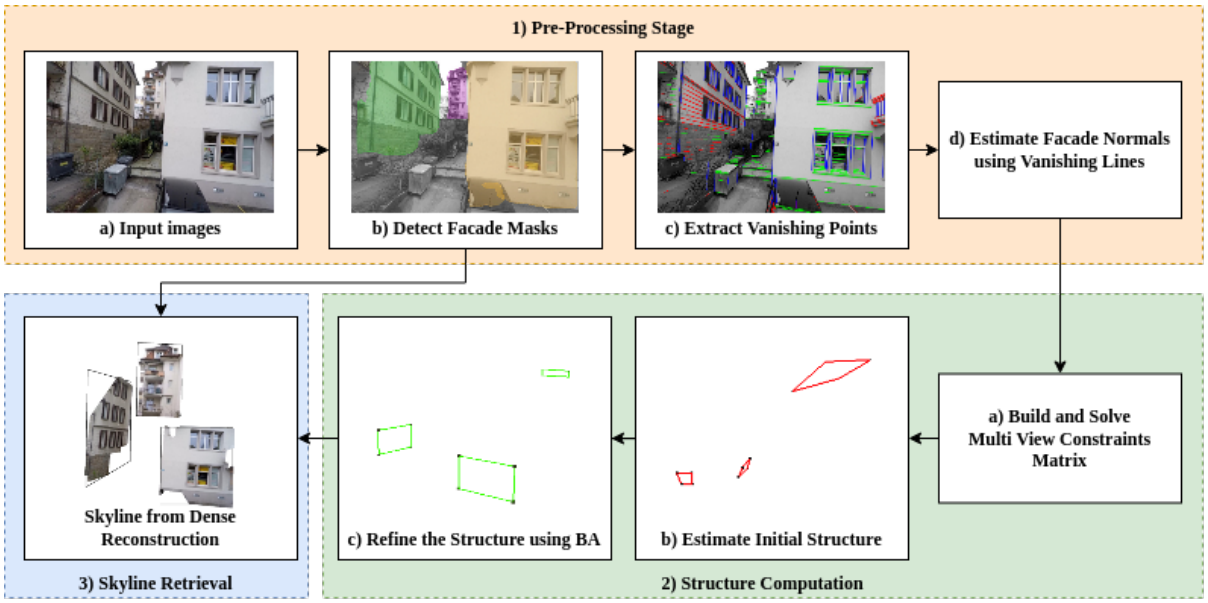


Figure 3.1: Overall pipeline of the proposed method. First four steps (in the top row) are part of the pre-processing stage (1). The next three steps (from right of bottom row) form the structure computation stage (2). And the last step represents the skyline retrieval stage (3). Images in each step represents the results obtained real world UrbanMAV dataset introduced by [35]. Step 1(b) shows the segmentation masks of the planar facades generated by PlaneRCNN. Step I(c) shows the line segments on the detected facades coloured by their vanishing point directions. Step 2(b) shows the initial estimate of the 3D line junctions. Step 2(c) shows the refined 3D structure after bundle adjustment using various geometric constraints. Step 3 shows the dense reconstruction and the retrieved skyline (black border surrounding each facade).

3.1 METHODOLOGY

3.1.1 Pipeline

The steps involved in the proposed pipeline are described as follows:

- Preprocessing of UAV images
- Initialization and refinement of sparse structure using geometric constraints
- Dense reconstruction followed by skyline retrieval

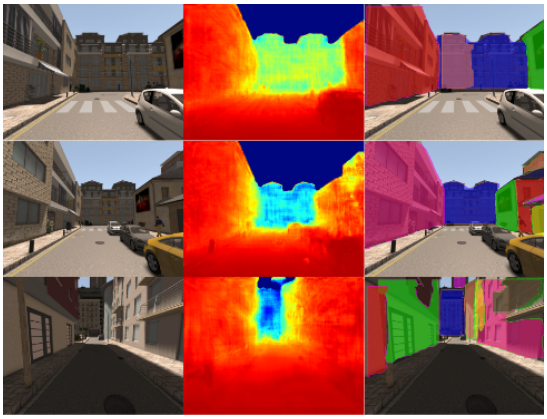


Figure 3.2: Results on test images after fine-tuning PlaneRCNN on SYNTHIA dataset

3.1.2 Pre-Processing

Each image in the sequence is pre-processed using the steps depicted in Figure 1. Each step is described in the following sections.

3.1.2.1 Facade Detection

In urban scenes, skyline is formed by planar facades of buildings. For computational purposes, the facades can be assumed to be planes in 3D.

To predict the facades/plane segment instances of buildings, we have trained PlaneRCNN on SYNTHIA dataset.

The architecture of PlaneRCNN is briefly described as follows. It uses Mask R-CNN [19] as its backbone to detect planar and non-planar regions, where each planar region is considered an object instance. Besides this, it contains two modules viz., segmentation refinement network, and warping loss module.

The segmentation refinement module jointly optimizes all the detected masks by comparing them using a cross-entropy loss. Its U-Net architecture [42] uses ConvAccu modules, which are based on non-local modules [48].

The warping loss module enforces the consistency of reconstructed 3D planes with a nearby view during the training. The 3D points p_n of a nearby view are projected on the current view, and current view coordinates p_c are read using bilinear interpolation. Then, p_c are transformed to nearby coordinate frame p_c^t to compute the L2 norm between p_c^t and p_n.

PlaneRCNN detects plane instances, predicts plane parameters, and per-pixel depthmap. However, we have observed that discontinuities like balconies, protrusions, and depression on the building walls lead to a poor prediction of plane normals and depth map. So, we limit its usage to predict plane segment masks.

Figure 3.3: A sample constraints matrix built from j^{th} view of a 3D quadrilateral defined by the vertices $\mathbf{V}_1$, $\mathbf{V}_2$, $\mathbf{V}_3$ and $\mathbf{V}_4$. $\mathbf{P}^j$ represents the projection matrix of j^{th} view. $\mathbf{t}_x$ and $\mathbf{t}_y$ represent the standard point triangulation constraints. $\mathbf{n}^j$ represents the normal of the plane computed using the equation (3.1). First eight rows of the matrix represent frustum constraints. Next eight rows are from the triangulation constraints. The last four rows arise from the orientation/normal constraints.

$$\begin{pmatrix}
l_1^{jT}\mathbf{P}^j & 0 & 0 & 0 & l_1^{jT}\mathbf{p}_4^j \\
0 & l_1^{jT}\mathbf{P}^j & 0 & 0 & l_1^{jT}\mathbf{p}_4^j \\
0 & l_2^{jT}\mathbf{P}^j & 0 & 0 & l_2^{jT}\mathbf{p}_4^j \\
0 & 0 & l_2^{jT}\mathbf{P}^j & 0 & l_2^{jT}\mathbf{p}_4^j \\
0 & 0 & l_3^{jT}\mathbf{P}^j & 0 & l_3^{jT}\mathbf{p}_4^j \\
0 & 0 & 0 & l_3^{jT}\mathbf{P}^j & l_3^{jT}\mathbf{p}_4^j \\
0 & 0 & 0 & l_4^{jT}\mathbf{P}^j & l_4^{jT}\mathbf{p}_4^j \\
l_4^{jT}\mathbf{P}^j & 0 & 0 & 0 & l_4^{jT}\mathbf{p}_4^j \\
\mathbf{t}_x^{1j} & 0 & 0 & 0 & t_{4x}^{1j} \\
\mathbf{t}_y^{1j} & 0 & 0 & 0 & t_{4y}^{1j} \\
0 & \mathbf{t}_x^{2j} & 0 & 0 & t_{4x}^{2j} \\
0 & \mathbf{t}_y^{2j} & 0 & 0 & t_{4y}^{2j} \\
0 & 0 & \mathbf{t}_x^{3j} & 0 & t_{4x}^{3j} \\
0 & 0 & \mathbf{t}_y^{3j} & 0 & t_{4y}^{3j} \\
0 & 0 & 0 & \mathbf{t}_x^{4j} & t_{4x}^{4j} \\
0 & 0 & 0 & \mathbf{t}_y^{4j} & t_{4y}^{4j} \\
\mathbf{n}^j & -\mathbf{n}^j & 0 & 0 & 0 \\
0 & \mathbf{n}^j & -\mathbf{n}^j & 0 & 0 \\
0 & 0 & \mathbf{n}^j & -\mathbf{n}^j & 0 \\
\mathbf{n}^j & 0 & 0 & -\mathbf{n}^j & 0
\end{pmatrix}
\begin{pmatrix}
\mathbf{V}_1 \\ \mathbf{V}_2 \\ \mathbf{V}_3 \\ \mathbf{V}_4 \\ 1
\end{pmatrix} = 0$$

3.1.2.2 Normal Estimation

Each facade contains a horizontal and a vertical vanishing point. A line joining any two vanishing points is called a Vanishing Line. Normal (**n**) of the facade plane can be computed using vanishing line (l_v) using the formula:

$$\mathbf{n} = \mathbf{R}^T(\mathbf{K}^T l_v) \tag{3.1}$$

where **R** and **K** represent the rotation and camera calibration matrices respectively.

LSD algorithm, as mentioned in [15] is used to extract the line segments within the facade segment in the image. Vanishing points are computed and are assigned to the line segments using the approach described in [27].

3.1.3 Initialization

Buildings can be reconstructed by considering the piece-wise planar surfaces of the facades. To achieve that, a 2D polygon (formed by joining the line junctions) is detected on the facade's image and tracked across neighbouring views. We use this information along with standard triangulation to build a multi-view constraints matrix. This matrix's null space is found using SVD (Singular Value Decomposition) to get an initial algebraic estimate of the structure. To minimize the residual errors in the initial estimate, another step of least-squares minimization is performed, which serves as the initialization for the 3D structure (depicted by red polygons in Fig 3.1).

At this stage, the 3D structure represents approximate positions of vertices of the 3D polygons on buildings' facades.

In the bundle adjustment stage (explained in the next section), different combinations of planar constraints are imposed besides the standard reprojection error (E_r). The constraints ensure that the vertices and the poses are simultaneously optimized while maintaining geometric consistency.

Frustum Constraint

A 3D polygon is represented by a list of vertices:

$$\begin{aligned}\mathbf{V}_1 &= (V_{1x}, V_{1y}, V_{1z})^T \\ \mathbf{V}_2 &= (V_{2x}, V_{2y}, V_{2z})^T \\ &\dots\end{aligned}$$

and so forth.

The edges of the polygon in the image may be denoted as $l_1, l_2, \dots, l_s$. If an edge i is projected from the center of the camera, it sweeps a plane in 3D.

The volume bounded by the planes formed by each edge of the polygon is defined as a frustum.

As each vertex of an s-sided polygon lies on the two intersecting edges, s vertices give rise to $2s$ frustum constraints from each view. If the polygon is visible in n views, each polygon gives rise to $2sn$ frustum constraints:

For a vertex $\widetilde{\mathbf{V}}_j$ (where represents homogenous coordinates) of an i^{th} quadrilateral visible in n views, the following equation represents the $2n$ frustum constraints.

$$\begin{aligned}(\mathbf{P}^{1T} l^1_{1i})^T \widetilde{\mathbf{V}}_j &= 0 \\ (\mathbf{P}^{1T} l^1_{2i})^T \widetilde{\mathbf{V}}_j &= 0 \\ &\vdots \\ (\mathbf{P}^{nT} l^n_{1i})^T \widetilde{\mathbf{V}}_j &= 0 \\ (\mathbf{P}^{nT} l^n_{2i})^T \widetilde{\mathbf{V}}_j &= 0\end{aligned}$$

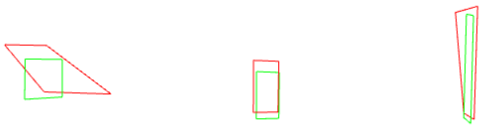


Figure 3.4: Results of the sparse structure before and after the modified bundle adjustment. Red polygons represent the estimate of the structure on a SYNTHIA sequence before the bundle adjustment step. Green polygons represent the refined structure after bundle adjustment using all the combinations of constraints mentioned in the section 3.3

where j represents all vertices lying on edge i.

The error term representing the frustum constraint, in general, can be defined as:

$$e_f = ||(\mathbf{P}^T l)\widetilde{\mathbf{V}}||_2 \tag{3.2}$$

Orientation Constraint

The normals of each facade are computed from the vanishing lines. Its error term can be represented as follows :

$$e_o = ||\mathbf{n}^T.(\mathbf{V}_2 - \mathbf{V}_1)||_2 \tag{3.3}$$

Scalar Triple Product (STP)

Scalar Triple Product represents the volume of a parallelopiped. The scalar triple product of four points in 3D is zero if they are coplanar. It is computed using the 3D vertices:

$$e_s = (\mathbf{V}_3 - \mathbf{V}_1).[(\mathbf{V}_2 - \mathbf{V}_1) \times (\mathbf{V}_4 - \mathbf{V}_1)] \tag{3.4}$$

Manhattan Constraint

Based on their normals, facades are assigned one of the two horizontal vanishing directions. As they are orthogonal to each other, the Manhattan constraint is defined as:

$$e_m = \mathbf{n}_1^T.\mathbf{n}_2 \tag{3.5}$$

Multi-view constraints matrix

It is to be noted that the Triangulation, Frustum, and Orientation constraints are linear in terms of the 3D vertices. So, a combination of these three constraints is used to build the multi-view constraints matrix shown in Figure (3.3).

(a)

(b)

Figure 3.5: a) Image of a real-world building captured by an RGB camera mounted on a UAV. b) shows a novel view of the dense reconstruction obtained with our method using only 5 views of the scene.

As it captures the constraints of all the 3D quadrilaterals obtained from all views, it is used to get a one-shot initialization of the all the piecewise planar structures in the scene.

As this is in the form $\mathbf{AX} = \mathbf{0}$, its null space (containing the vertices) is solved using SVD. This gives the initial algebraic estimate of the structure.

3.1.4 Modified Bundle Adjustment

So far, the vertices obtained may not lie on a plane, thereby deforming the planar structure of the facades of the building. To overcome this, we modify bundle adjustment to simultaneously optimize for the planar structure and the poses using different combinations of residual terms besides the reprojection error. The residuals are computed from the parameters viz., initialized 3D vertices, and the poses.

Total Residual

The total residual is computed as the weighted sum of all the constraints involved in the combination.

$$e = e_r + e_f + e_o + e_m + e_s \tag{3.6}$$

Figure 3.4 shows the results of the structure before and after the bundle adjustment.

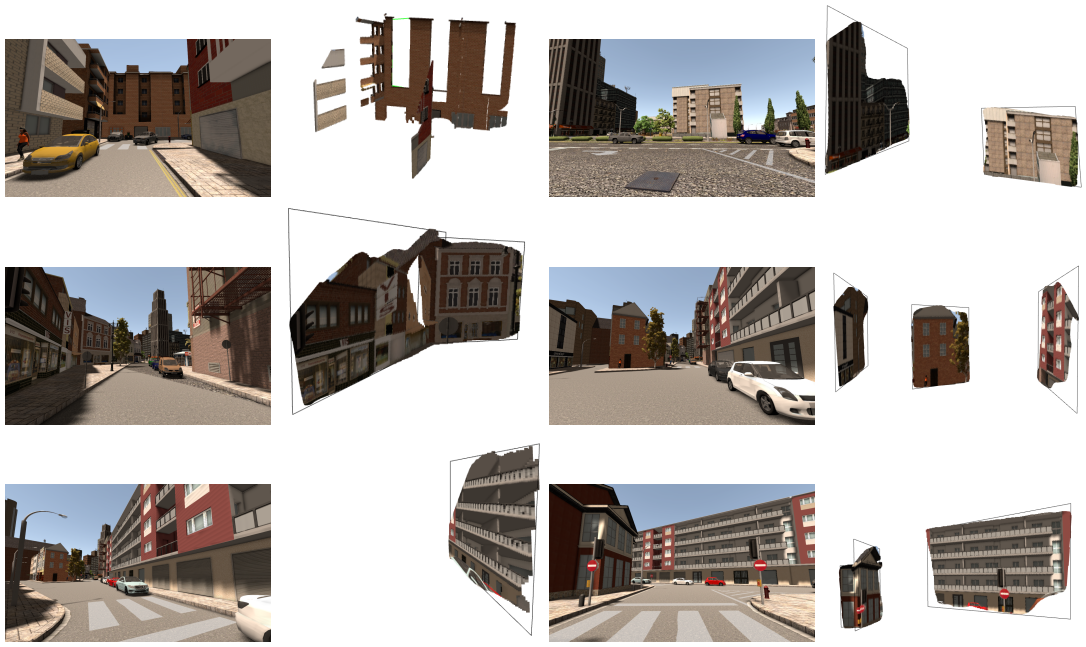


Figure 3.6: Qualitative results obtained on photo-realistic views from SYNTHIA dataset. Only 3-5 images per scene are considered. Skylines of the buildings retrieved using our method from 6 different scenes are shown. Dense reconstruction is visualized using Open 3D [53] software.

Table 3.1: Ablation study on contributions of two components in PlaneRCNN. VOI, RI and SC are the metrics with abbreviations Variation Of Information, Rand Index and Segmentation Covering respectively. Plane segmentation metrics are computed on the SYNTHIA dataset.

Method	VOI ↓	RI ↑	SC ↑
warping + refine (pre-trained)	2.556	0.536	0.376
warping + refine (re-rained on SYNTHIA)	1.495	0.793	0.677
refine (re-trained on SYNTHIA)	**1.222**	**0.863**	**0.770**

3.1.5 Skyline Retrieval

The plane parameters of each facade are computed from the refined structure. All pixels that lie inside the facade masks are then projected onto their respective reconstructed planes to generate a dense reconstruction of the plane segment. Thus the skyline is drawn along the border of the plane segment.

3.2 EXPERIMENTS AND RESULTS

We have evaluated the proposed pipeline on different scenes of SYNTHIA as well as UrbanMAV datasets.

Table 3.2: Various errors (averaged across the test scenes) in the initial estimates of the structure after the stages 2a and 2b. This shows the improvement in structure due to least squares minimization or geometric correction (stage 2b) on the initial estimate obtained algebraically using SVD of the multi-view constraints matrix (stage 2a).

Stage of initialization	Plane Orientation (deg)	Depth (cm)	Coplanarity (cm^3)	Manhattan (deg)
2a	18.183	14.227	0.935	31.798
2b	**11.079**	**13.825**	**0.697**	**31.244**

Table 3.3: Ablation study of imposing various structural constraints on BA (stage 2c of Figure 1), showing the overall improvement in the structure. The errors are computed using the ground truth of SYNTHIA dataset (Averaged across the test scenes).

Constraints	Plane Orientation (deg)	Depth (cm)	Coplanarity (cm^3)	Manhattan (deg)
e_r	10.824	12.884	1.093	30.905
$e_r + e_s$	10.545	**12.737**	1.042	30.793
$e_r + e_s + e_f$	10.075	52.522	1.269	33.121
$e_r + e_s + e_f + e_m$	**9.781**	21.799	**0.903**	**29.156**

We have used PlaneRCNN [30] framework for building facade/plane segmentation. Originally, PlaneRCNN is trained on the ScanNet [8], an indoor dataset. This resulted in poor predictions when using the pre-trained model SYNTHIA [43], which is an outdoor dataset. Two problems were identifed with the pre-trained model detections - 1) Only planes near the camera were detected, and 2) It detected planes of other objects like cars, which are not required. As shown in Figure(3.2), re-training the network on SYNTHIA resulted in better mask predictions. The re-trained model is able detect far-off and very small planes while ignoring unnecessary planes like the ground, car roof, car doors, etc.

We have trained PlaneRCNN on 1018 images of the Synthia-SUMMER-04 sequence for ten epochs with a learning rate of $1e^{-4}$ on an NVIDIA GTX 1080 Ti GPU. SYNTHIA dataset does not have plane instance segmentation. So, the dataset was manually annotated with plane instances. We have observed that, PlaneRCNN with the segmentation refinement module gives a better result than with both the refinement and the warping module in detecting the plane segments on the facades of buildings. We have evaluated plane segmentation quality for the pre-trained model and trained model (using different modules) on the SYNTHIA dataset (tabulated in Table 2). The metrics used are - variation of information (VOI), Rand index (RI), and segmentation covering (SC) [50].

Variation of Information (VOI) metric is used for clustering comparison. It measures the distance between two segmentations in terms of their average conditional entropy and is given by

$$VOI(S, S') = H(S) + H(S') - 2I(S, S') \tag{3.7}$$

where, H and I represent, respectively, entropies and mutual information between two clusterings of data S and S'. Less similar the two segmentations, higher is the VOI.

Rand Index allows the comparison of a test segmentation with multiple ground-truth segmentations through soft nonuniform weighting of pixel pairs as a function of the variability in the ground-truth set.

$$RI(S, \{G_k\}) = \frac{1}{T} \sum_{i<j} [c_{ij} p_{ij} + (1 - c_{ij})(1 - p_{ij})] \tag{3.8}$$

where, $\{G_k\}$ is the set of ground-truth segmentations, c_{ij} is the event that pixels i and j have same label and p_{ij} is its probability and T is total number of pixel pairs.

Segmentation Covering (SC) of a segmentation S by a segmentation S' is defined as

$$C(S' \rightarrow S) = \frac{1}{N} \sum_{R \in S} |R| \cdot \max_{R' \in S'} O(R, R') \tag{3.9}$$

where,

$$O(R, R') = \frac{|R \cap R'|}{|R \cup R'|} \tag{3.10}$$

and N denotes total number of pixels in the image. Similarly, the covering of a machine segmentation S by a family of ground-truth segmentations $\{Gi\}$ is defined by first covering S separately with each human segmentation G_i and then averaging over the different humans. To achieve perfect covering, the machine segmentation must explain all of the human data. We can then define two quality descriptors for regions: the covering of S by $\{G_i\}$ and the covering of $\{G_i\}$ by S.

We have used C++ based optimization library `ceres-solver`[2] for initialization and bundle adjustment steps.

Figure (3.5b) shows the reconstruction of a real-world building captured from a UAV obtained using our proposed method. The UAV used is equipped with an mvBlueFOX monocular camera with a resolution of 1.3MP. Five images of the scene have been used to reconstruct the scene using the proposed pipeline.

Figure (3.6) shows the qualitative results with the dense reconstructions and skylines obtained by following our method.

Quantitative results obtained on the synthetic dataset (SYNTHIA) only are reported in tables 2 3, as the ground truth depth maps are unavailable on real world UrbanMAV dataset. The error metrics are computed by comparing the estimated vertices against the vertices obtained from ground truth depth. The values are presented in (cm) to show the accuracy of the method when evaluated in synthetic scenes. Plane Orientation and Manhattan errors depicts the average deviation of the normals of the planes in scene w.r.t the ground truth plane normals. Table 2 shows the advantage of using stage 2b to improve the initial algebraic estimate. Table 3 shows the improvement in the structure after the bundle adjustment modified by imposing various constraints. As the orientation residual term (e_o) in BA is computed using the normals computed from the vanishing line, it is sensitive to the errors in the line feature detection. As this resulted in increase of the error metrics after the BA stage, it is not reported in Table 3. Nonetheless, the Orientation constraint in the stages 2a and 2b, proved to be useful in reducing the plane normal deviation of estimated structure.

3.3 CONCLUSION

In this chapter, we have shown that using the constraints have improved the depth and orientation estimates of piecewise planar structures in city scale urban environments.

By training PlaneRCNN to detect the buildings' planar facades, the geometric information of each visible facade can be extracted. Imposing the deduced multi-view geometric constraints by modifying the standard bundle adjustment resulted in improved depth and orientation estimates. The dense reconstruction of the facades is obtained by using the facade masks generated by the neural network. In some cases, the increase in depth error has been compensated by the decrease of orientation error, ensuring structural improvement.

The skyline, thus retrieved from the dense reconstruction, can be used in navigation and path planning.

ACKNOWLEDGEMENTS

We thank Shivaan Sehgal and Sidhant Subramanian, for annotating the building facades in SYNTHIA dataset and Mukul Khanna, for helping out with facade detection network experiments. We also thank Krishna Murthy J. at Real and Embodied AI Lab, Université de Montréal for valuable feedback/advice during the brainstorming sessions.

Chapter 4

A new geometric approach for three view line reconstruction and motion estimation in Manhattan Scenes

Structure and motion estimation from images are the quintessential steps for state estimation of a robot equipped with a monocular camera. Map of an environment is essential for navigation of a robot. Map generated using line features is structurally richer than that of point features. Even though dense methods exist, they are computationally expensive and are not suitable for real time tasks. Vanishing points, when extracted from the line features in Manhattan scenes, can be used to directly compute the relative rotation between two views.

In spite of these advantages the approaches using line features are more prone to degenerate cases [52]. Even though numerous approaches for line based reconstruction exist in the literature, most of them rely on point-based methods to initialize the poses. Performance of the methods that rely on point features can degrade in scenes with low-texture and in the presence of occlusions like trees, etc., Recent works like [12][29][57] therefore leverage the structural regularity in Manhattan scenes to simplify the structure and motion estimation.

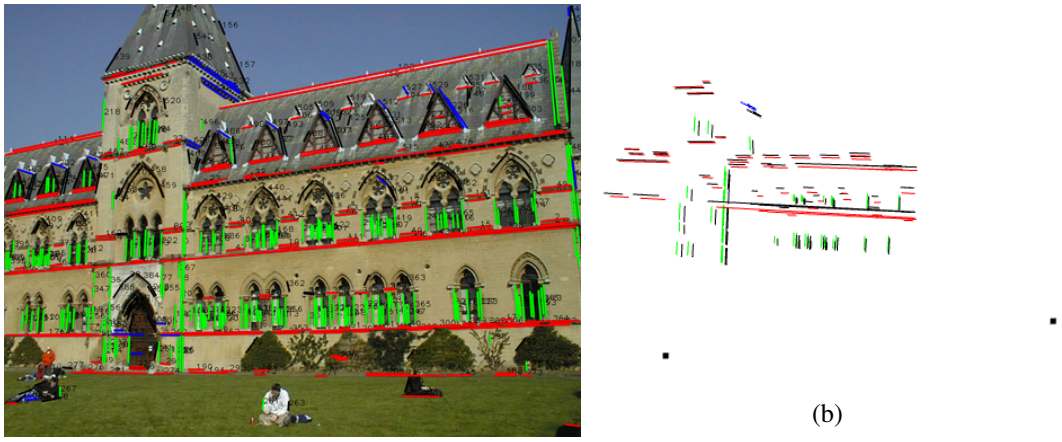
Our contributions in this chapter are as follows:

- a new geometric approach for joint estimation of three view structure and motion from line features in Manhattan scenes with mathematical elegance
- 1-parameter based cost function that allows the use of hyper edge in the factor graph optimization to increase robustness to correspondence outliers

The chapter is organized as follows. Related Works are covered in Section II. Methodology of the proposed approach is presented in Section III. Experimental results are then presented in Section IV. Finally, we conclude the chapter and state the scope for future work in Section V.

4.1 Methodology

In this section, the details of each stage in the pipeline as shown in figure 2 are described.



(a) (b)

Figure 4.1: a) Real world image from University Library scene of VGG MultiView dataset. Figure b) shows the reconstructed 3D lines and recovered camera poses (marked as black dots) using our approach. The lines are coloured Red, Green and Blue as per their vanishing point direction. Black lines represent the ground truth 3D lines.

4.1.1 Overview

Each each image is pre-processed to extract the line features and to compute the vanishing point directions. Then relative rotation matrices are computed using vanishing points. 3D lines and relative translations are estimated by substituting the computed rotation matrices. Then a modified bundle adjustment is performed to minimize the error residuals and to optimize the initialized structure as well as camera poses.

Before we describe the details of our approach, 3D line representation and the notation used is presented. As suggested in [5], we use Plucker representation for 3D line projection and orthonormal representation for update step during the optimization. Plucker coordinates of a 3D line are represented by $(\boldsymbol{d}, \boldsymbol{m})$, where $\boldsymbol{d}$ represents the direction of the 3D line. In our approach, $\boldsymbol{d}$ is directly set to the vanishing point direction. And $\boldsymbol{m}$ represents the moment vector, which is perpendicular to the plane swept by back-projecting the 2D line from the camera center. It can be computed using the 2D line as

$$\hat{\boldsymbol{m}} = \boldsymbol{\mathcal{K}}^{-1}\boldsymbol{l} \tag{4.1}$$

Here $\hat{\boldsymbol{m}}$ is the unit direction of the moment vector (therefore, $\boldsymbol{m} = m\hat{\boldsymbol{m}}$). $\boldsymbol{l}$ is the observed 2D line. $\boldsymbol{\mathcal{K}}$, the line projection matrix of camera with focal lengths (f_u, f_v)and image center (c_u, c_v) is given as

$$\boldsymbol{\mathcal{K}} = \begin{bmatrix} f_v & 0 & 0 \\ 0 & f_u & 0 \\ -f_v c_u & f_u c_v & f_u f_v \end{bmatrix} \tag{4.2}$$

In the three view formulation that we describe in the following sections, it is assumed that poses of views 2 and 3 are considered in the coordinate frame of camera 1. This helps in direct substitution of the relative rotations in line reprojection as well the trifocal constraints.

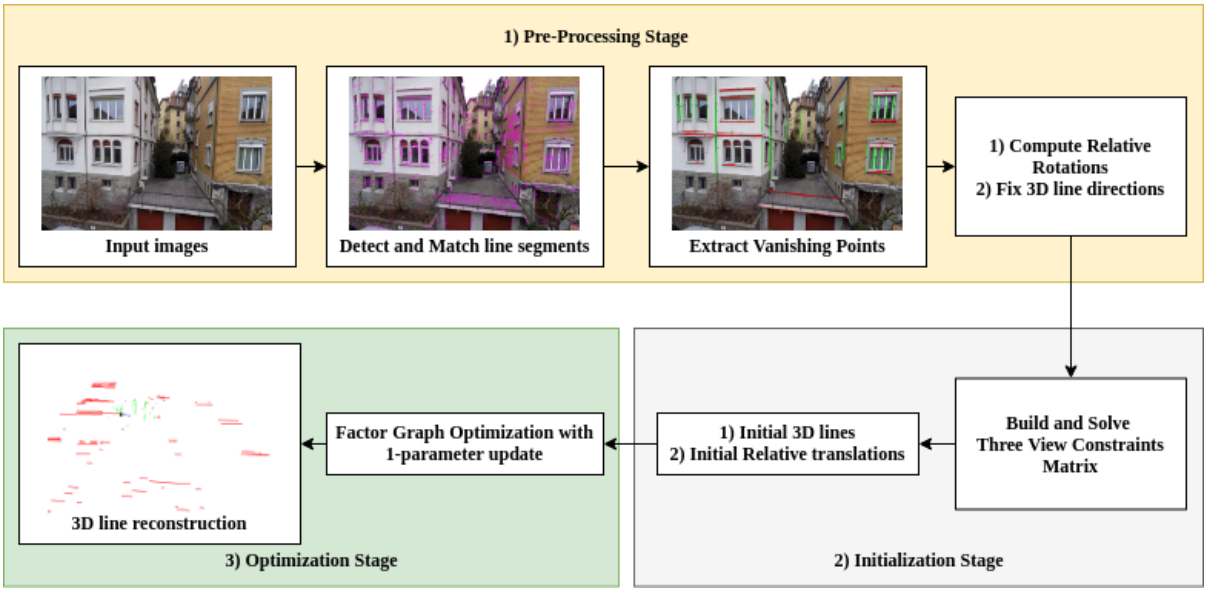


Figure 4.2: Block diagram of our approach. It consists of three stages viz., Pre-Processing, Initialization and Optimization stage. The vanishing points extracted in each image from the detected line segments are used for computing relative rotations and for fixing the direction component of 3D lines. As described in Section IIID, the constraints matrix is then built and solved to get initial structure and motion estimate. This is further optimized by 1-parameter line BA in a factor graph formulation.

4.1.2 Pre-processing

- Given three images of a Manhattan scene, line segments are firstly detected using LSD algorithm.
- 2D line segments are matched across the three views using Line Binary Descriptor (LBD) method.
- Also, the vanishing points of the scene are extracted from the line segments in each image using the method described in [34].
- Also, compute the vanishing lines for each pair of vanishing points of each image.
- Normals of the surfaces along the two horizontal vanishing point directions are computed from the vanishing line $\boldsymbol{l}_v$ using equation 1.

4.1.3 Rotation Estimation

The relative rotations are estimated from the normals. The axis of rotation is computed as the cross-product of the corresponding normals in View 1 and View 2. The angle of rotation θ is computed using the cosine-formula for dot product. The estimated rotations are then optimised to minimizing the rotation error between the normals across the three views.

4.1.4 Constraints Matrix

The optimized rotations are then substituted into the line reprojection and trifocal constraint equations. This gives rise to a linear set of constraints in terms of relative translations and the 3D line parameters.

4.1.4.1 3D Line reprojection constraint

The projection of a 3D line $(\boldsymbol{m}, \boldsymbol{d})$ onto a camera is given by the equation

$$\boldsymbol{l} = m\boldsymbol{\mathcal{K}}\boldsymbol{R}\hat{\boldsymbol{m}} + \boldsymbol{\mathcal{K}}[\boldsymbol{t}]_{\times}\boldsymbol{R}\boldsymbol{d} \tag{4.3}$$

where $m = |\boldsymbol{m}|$, magnitude of the moment vector and $\boldsymbol{R}, \boldsymbol{t}$ are rotation matrix and translation vector of the camera respectively.

Line reprojection constraints for a given 2D line segment measurement with end point $\boldsymbol{x}$ is defined as

$$\boldsymbol{x}^T\boldsymbol{l} = 0 \tag{4.4}$$

In the line projection equation, $\boldsymbol{\mathcal{K}}$ is known, as we consider a calibrated camera, $\boldsymbol{R}$ is computed using the normal vectors as mentioned in Section IIIC, the line direction $\boldsymbol{d}$ is given by the vanishing point direction, the unit moment vector $\hat{\boldsymbol{m}}$ can be computed using the equation (1).

Equation (4) therefore can be rearranged as

$$\boldsymbol{x}^T \begin{bmatrix} \boldsymbol{\mathcal{K}}\hat{\boldsymbol{m}}' & -\boldsymbol{\mathcal{K}}[\boldsymbol{d}']_{\times} \end{bmatrix} \begin{bmatrix} m \\ \boldsymbol{t} \end{bmatrix} = 0 \tag{4.5}$$

where $\hat{\boldsymbol{m}}' = \boldsymbol{R}\boldsymbol{m}$ and $\boldsymbol{d}' = \boldsymbol{R}\boldsymbol{d}$.

4.1.4.2 Trifocal constraint for lines

The trifocal constraint in terms of the observations $\boldsymbol{l}_0, \boldsymbol{l}_1, \boldsymbol{l}_2$ of a 3D line in three views is given by the equation, [18]

$$[\boldsymbol{l_1}^T[\boldsymbol{T}_1, \boldsymbol{T}_2, \boldsymbol{T}_3]\boldsymbol{l_2}][\boldsymbol{l_0}]_{\times} = \boldsymbol{0}^T \tag{4.6}$$

where,

$$\boldsymbol{T}_i = \boldsymbol{r}_i'\boldsymbol{t}''^T - \boldsymbol{t}'\boldsymbol{r}_i''^T \tag{4.7}$$

is the component of the Trifocal tensor. $\boldsymbol{r}_i'$ and $\boldsymbol{r}_i''$ are the column vectors of the relative rotation matrices $\boldsymbol{R}'$ and $\boldsymbol{R}''$ of second and third views respectively. Similarly $\boldsymbol{t}'$ and $\boldsymbol{t}''$ are the relative translation vectors of second and third views respectively.

It can be observed that the trifocal constraint is the function of line observations and the relative poses of second and third cameras. Therefore, by substituting the relative rotations and by replacing $\boldsymbol{l}_0 = (a_0, b_0, c_0)$, equation (6) can be rewritten as a linear function of relative translations as follows

$$\begin{bmatrix} \boldsymbol{x}_s'^T \mathcal{K} \hat{\boldsymbol{m}}' & 0 & -\boldsymbol{x}_s'^T \mathcal{K} [\boldsymbol{d}']_\times \\ \boldsymbol{x}_e'^T \mathcal{K} \hat{\boldsymbol{m}}' & 0 & -\boldsymbol{x}_e'^T \mathcal{K} [\boldsymbol{d}']_\times \\ 0 & \boldsymbol{x}_s''^T \mathcal{K} \hat{\boldsymbol{m}}'' & -\boldsymbol{x}_s''^T \mathcal{K} [\boldsymbol{d}'']_\times \\ 0 & \boldsymbol{x}_e''^T \mathcal{K} \hat{\boldsymbol{m}}'' & -\boldsymbol{x}_e''^T \mathcal{K} [\boldsymbol{d}'']_\times \\ 0 & c_0(\boldsymbol{l}_1^T \boldsymbol{r}_2') \boldsymbol{l}_2^T - b_0(\boldsymbol{l}_1^T \boldsymbol{r}_3') \boldsymbol{l}_2^T & -c_0 \boldsymbol{l}_1^T (\boldsymbol{r}_2''^T \boldsymbol{l}_2) + b_0 \boldsymbol{l}_1^T (\boldsymbol{r}_3''^T \boldsymbol{l}_2) \\ 0 & -c_0(\boldsymbol{l}_1^T \boldsymbol{r}_1') \boldsymbol{l}_2^T + a_0(\boldsymbol{l}_1^T \boldsymbol{r}_3') \boldsymbol{l}_2^T & c_0 \boldsymbol{l}_1^T (\boldsymbol{r}_1''^T \boldsymbol{l}_2) - a_0 \boldsymbol{l}_1^T (\boldsymbol{r}_3''^T \boldsymbol{l}_2) \\ 0 & b_0(\boldsymbol{l}_1^T \boldsymbol{r}_1') \boldsymbol{l}_2^T - a_0(\boldsymbol{l}_1^T \boldsymbol{r}_2') \boldsymbol{l}_2^T & -b_0 \boldsymbol{l}_1^T (\boldsymbol{r}_1''^T \boldsymbol{l}_2) + a_0 \boldsymbol{l}_1^T (\boldsymbol{r}_2''^T \boldsymbol{l}_2) \end{bmatrix} \begin{bmatrix} m \\ \boldsymbol{t}'' \\ \boldsymbol{t}' \end{bmatrix} = 0 \tag{4.8}$$

Figure 4.3: A sample constraints matrix of a 3D line which is our main contribution, enables joint estimation of the two relative translations and 3D line depth parameter using a three view geometry for Manhattan scenes. First four rows represent the standard line triangulation constraints. Last three rows are from the trifocal constraints. All the constraints arising from all the 3D lines are stacked onto a single matrix and solved for its null space using Singular Value Decomposition (SVD).

Substituting the 3D line direction vector $\hat{\boldsymbol{d}}$ and the unit moment vector $\hat{\boldsymbol{m}}$, the reprojection constraints contain only one unknown parameter m i.e., the magnitude of the moment vector which represents the perpendicular distance of the 3D line from the camera origin.

The substituted constraints for both start and end points of the line segments (represented by $\boldsymbol{x_s}$ and $\boldsymbol{x_e}$) are then stacked in to a matrix, whose null space gives the least squares estimate of the 3D line distance (or the magnitude of the moment vector) and the two relative translations. A sample constraints matrix is shown in 4.3

The covariance of the computed solution can be computed by the formula

$$\boldsymbol{\Omega} = \boldsymbol{C}^T \boldsymbol{C}$$

where C is the constraints matrix with mean-centered and unit normalized columns.

4.1.5 Factor Graph based Optimization

Three view reprojection error is computed as a 6-vector $\boldsymbol{e}_t^i = [\boldsymbol{e}_{l1}^{i\mathrm{T}}, \boldsymbol{e}_{l2}^{i\mathrm{T}}, \boldsymbol{e}_{l3}^{i\mathrm{T}}]^{\mathrm{T}}$, where $\boldsymbol{e}_{lk}$ is the line reprojection error in k^{th} view, computed as:

$$\boldsymbol{e}_l = \mathrm{d}\,(\boldsymbol{z},\, \boldsymbol{l}') = \left[\frac{\boldsymbol{x}_s^{\mathrm{T}} \boldsymbol{l}'}{\sqrt{l_1^2 + l_2^2}},\ \frac{\boldsymbol{x}_e^{\mathrm{T}} \boldsymbol{l}'}{\sqrt{l_1^2 + l_2^2}} \right]^{\mathrm{T}}$$

e_t^i contains four variables/nodes viz., three SE3 nodes (one for each view) and one 3D line node. As per our knowledge, only **g2o** supports hyper-edges (edges that bind more than two nodes). So, a hyper-edge is added to the factor graph for each triplet hypothesis.

Total cost function is given as (in the switchable constraint form)

$$\boldsymbol{e}_{total} = \sum_{i=1}^{n} \left(\boldsymbol{e}_t^{i\mathrm{T}} \boldsymbol{\Omega}_i^{-1} \boldsymbol{e}_t^i \right)$$

Table 4.1: Our method vs Baseline [52] on VGG and SYNTHIA datasets. (r - reprojection constraint. t - trifocal constraint)

Dataset	VGG		SYNTHIA	
Metric	$e_{depth}(m)$	$e_{dir}(m)$	$e_{depth}(m)$	$e_{dir}(m)$
Ours (r+t)	**1.41**	**0.17**	**0.65**	**0.41**
Ours (r)	2.92	0.26	0.72	0.48
Baseline (gt) [52]	5.88	0.70	7.44	0.78
Baseline (estim) [52]	6.67	0.85	8.29	0.76

Table 4.2: Comparision of robustness to correspondence outliers

Outlier (percentage)	Baseline [52]		Ours	
	$e_{depth}(m)$	$e_{dir}(m)$	$e_{depth}(m)$	$e_{dir}(m)$
0	5.21	0.61	**1.53**	**0.15**
20	6.25	0.69	**3.73**	**0.21**
40	6.30	0.83	**4.64**	**0.43**

where n is the number of 3D lines.

The directions of the 3D lines are fixed in the optimization of the 3D line and camera poses. Additionally, instead of representing the reprojection constraint of each view as an edge in the factor graph, a hyper edge is used to incorporate the reprojection constraint of the 3D line onto corresponding 2D line observations in three views. This helps in increase of robustness, in the presence of outliers.

4.1.6 Plucker line trimming

For final reconstruction, the optimized 3D plucker lines are trimmed using the formula given in [51] to compute the optimal 3D line segment end points using the 2D line segment end points.

4.2 Experiments

In this section, we describe the experiments performed to evaluate the proposed method. We have implemented our approach in C++ libraries Eigen [16] and g2o [25] for matrix operations and factor graph optimization steps respectively.

4.2.1 Datasets

The proposed approach is evaluated on 1 synthetic dataset SYNTHIA [44] and 2 real world datasets VGG [49], UrbanMAV [36].

Table 4.3: Evaluation in the presence of line segment end point noise

End point noise (px)	Baseline [52]		Ours	
	$\mathrm{e}_{depth}(m)$	$\mathrm{e}_{dir}(m)$	$\mathrm{e}_{depth}(m)$	$\mathrm{e}_{dir}(m)$
0	4.35	0.51	**0.77**	**0.13**
0.25	5.28	0.62	**1.85**	**0.26**
0.5	5.66	0.69	**2.30**	**0.31**

4.2.1.1 VGG MultiView dataset

It contains 3 real world scenes. For each scene in the dataset, 3 images along with their ground truth camera poses, 2D line segments, their correspondences and the ground truth 3D lines are provided.

4.2.1.2 SYNTHIA

It is a synthetic dataset, with a sequence of images from the view point of a camera mounted on a car navigated in a virtual city.

4.2.1.3 Urban MAV

This dataset contains sequence of images captured in the streets of Zurich city by a monocular camera mounted on a drone flying at low altitudes.

4.2.2 Evaluation Metrics

4.2.2.1 3D line depth error

$$e_{depth} = ||\boldsymbol{x}_{gt}^{mid} - \boldsymbol{S}(\boldsymbol{x}_{estim}^{mid})||$$

where S represents the similarity transformation between $\boldsymbol{x}_{est}^{mid}$, the mid points of estimated 3D line segments, and $\boldsymbol{x}_{gt}^{mid}$, the mid points of ground truth 3D line segments.

4.2.2.2 3D line direction error

$$e_{dir} = ||\boldsymbol{d}_{gt} - \boldsymbol{d}_{est}||$$

where $\boldsymbol{d}_{gt}$ represents the normalized direction of the ground truth 3D line and $\boldsymbol{d}_{est}$ represents the normalized direction of the estimated 3D line segments.

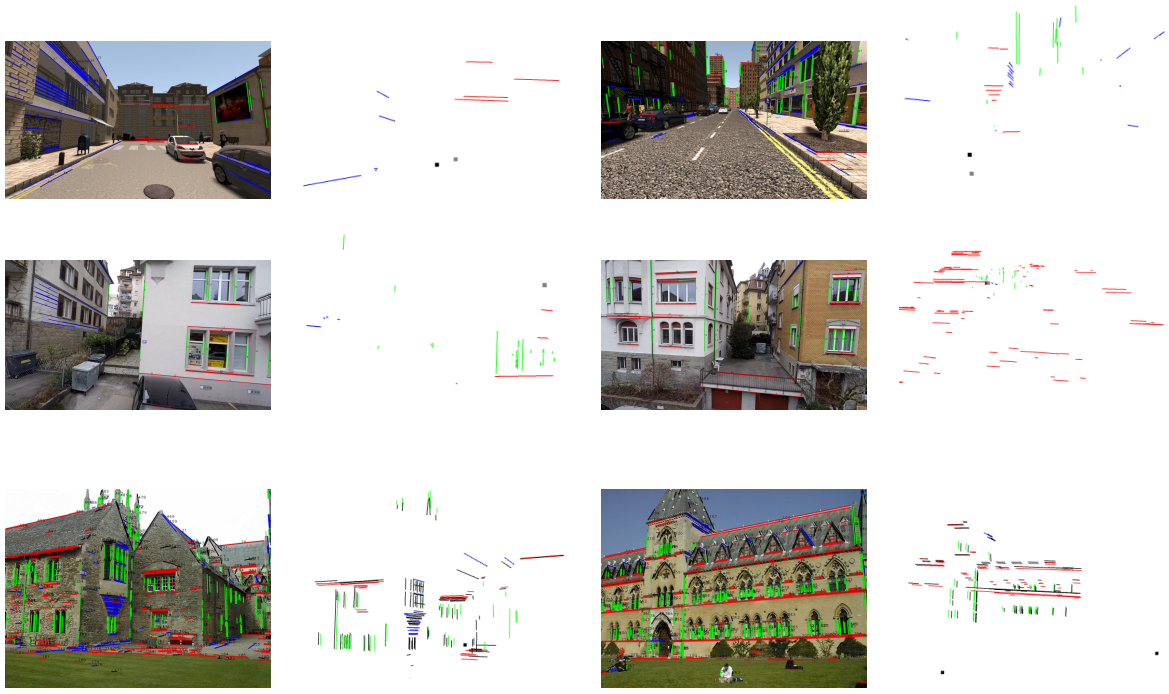


Figure 4.4: 3D line reconstructions obtained by using our method across three different datasets. First row shows the reconstruction on SYNTHIA which is a synthetic one. Second and third row shows the 3D line reconstruction on UrbanMAV and VGG MultiView datasets respectively.

4.2.3 Quantitative Results

For quantitative evaluation, we consider standard 3D line triangulation as mentioned in [52] as the baseline.

4.2.3.1 Our method vs Baseline

As the baseline approach requires known poses of the three cameras, two variants of the baseline exists for comparison. We report the results of baseline approach using ground truth (gt) as well as estimated (estim) poses. We also compare our method with and without trifocal constraint in the initialization step.

Estimated 3D line segments using each method mentioned above are compared against the ground truth 3D line segments. We perform the estimation on all the 3 scenes of VGG MultiView dataset and 10 scenes of SYNTHIA dataset. The results are reported in Table I.

4.2.3.2 Correspondence outliers

Each scene in VGG has 100 lines with correct matches. The robustness of our method is evaluated by introducing correspondence outliers. In a trial, N random 2D lines in view 1 are chosen and are

matched against random 2D lines in views 2 and 3. Each trial is repeated for 100 times. Dynamic Covariance Scaling (DCS) proposed by [1] is used as robust kernel. The results are reported in Table II for N = 0, 20 and 40.

It is observed that our method performs better when compared against the baseline even with 1-parameter optimization for each 3D line. This is due to the fact that, disabling hyper edges connected to 3D line, with high covariance in the factor graph optimization, nullifying the gradients from the measurements of an outlier line.

4.2.3.3 2D line segment end point noise

2D line features are detected from the edges and therefore, the end points are prone to noise. As each 2D line segment has 2 end points, the Table III shows the results obtained by adding Gaussian noise with standard deviation 0, 0.25 and 0.5 pixels to all 2D line segment end points in each image.

In the case of baseline or other approaches, the line segment endpoint noise strongly effects the 3D line direction. The results show that our approach is weakly effected, as we fix the direction component of the 3D line. It is to be noted that the observed direction error is due to the vanishing point direction alone.

4.2.4 Qualitative Results

We report the 3D line segment reconstructions across 3 different datasets viz., VGG MultiView, SYNTHIA and UrbanMAV. Figure 4 shows the reconstruction results in three views for each scene.

4.3 Summary

In this chapter we have proposed a simple approach for three view structure and motion estimation from line features in Manhattan scenes. As a consequence of the mathematical elegance of the proposed approach, our method is simple to implement and robust to correspondence outliers as well as line segment end point noise.

Structure estimated using the proposed approach can be extended to higher level features like fitting planes for facade reconstruction of buildings, etc. for future work. The motion estimation if extended can be used in a SLAM system. To operate in the metric scale, integrating it within a visual inertial system can be tried.

Chapter 5

UrbanFly: Uncertainty-Aware Planning for Navigation Amongst High-Rises with Monocular Visual-Inertial SLAM Maps

With the increasing possibility of Urban Air Mobility (UAM) in the recent future, quadrotors' need to navigate through urban high-rises becomes increasingly inevitable. Monocular vision continues to be a widespread perception modality for quadrotors considering payload, portability, and endurance. Hence navigation based on monocular SLAM maps becomes a natural choice. However, these maps tend to be sparse, noisy, and not represented in the scale of the actual world. We can use the popular Visual Inertial SLAM [40] to obtain the quadrotor odometry and consequently bring the maps to metric scale. Yet, the sparsity of the point clouds are not entirely alleviated. Unfortunately, existing algorithms for planning are not designed to work with systems that rely on monocular vision-based perception. For example, works like [22], [59], either assume that the obstacle geometries are precisely known or rely on depth sensors to obtain obstacle geometries during run-time.

In this chapter, we present UrbanFly, a novel planning approach that can leverage the capabilities of monocular vision-based perception and at the same time counter the effect of underlying sparsity. Our approach has two core components: (i) a cuboid representation of the environment and (ii) trajectory optimizer specifically designed to exploit the chosen representation. The novelties and benefits of our approach are summarized below.

Algorithmic Contribution: We segment the high-rises in the image into their planar constituents and consequently build a cuboid representation of the obstacles (skyscrapers). We propose a trajectory optimizer capable of minimizing this estimate along with the conventional smoothness costs. It computes trajectories that avoids inflated version of the cuboid obstacles using a sequential convex programming (SCP) based optimizer. Importantly, the SCP is initialized based on safety estimates on a set of randomly drawn trajectories.

State-of-the-Art Performance:

- Our work is one amongst the few rare papers to demonstrate robust and collision-free autonomous flight with a single camera system

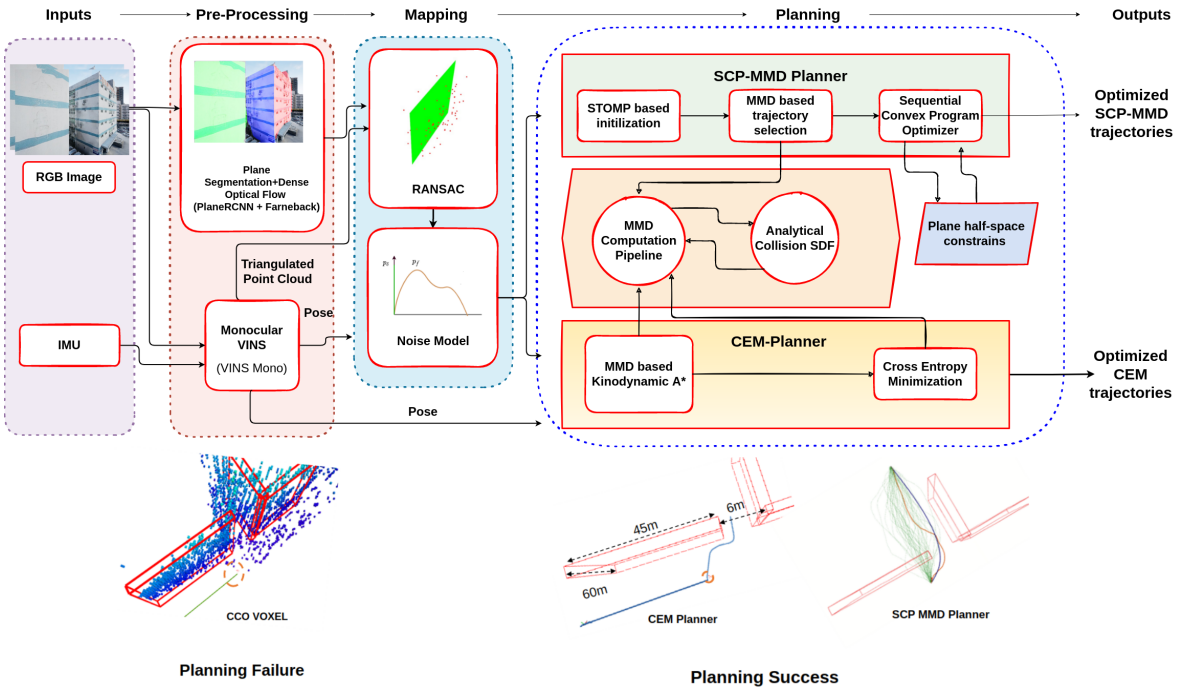


Figure 5.1: **Overview**: Accurate distance measurement to the closest obstacle and associated gradient cannot be expected from a noisy and sparse voxel grid representation built from the triangulated point cloud of a monocular visual-inertial SLAM (VINS) system. This in turn, poses a critical challenge for trajectory planning. As a potential solution, we propose UrbanFly: a pair of trajectory optimizers for safe navigation amongst high-rises in urban environments. Unlike active SLAM or feature-driven planning methods, our objective is not to tightly couple perception and planning. Instead, we intend to accommodate the capabilities and limitations of SLAM within trajectory optimization. To this end, we use the VINS back-end to construct a geometric representation of obstacles as cuboids and estimate the associated uncertainty in orientation and size. Our trajectory optimizers based on Cross-Entropy Method (CEM) and sequential convex programming achieve real-time performance by leveraging fast estimates of collision probability made possible by the geometrical representation of the obstacles. The figure also demonstrate a case where the baseline [17] that plans to compute trajectory fails whereas our method can successfully plan. Further details related to the figure is described in section 5.1.1

- We show advantages of our chosen cuboid representation of the urban-rise environment as compared to the more common voxel-grid representation. These specific advantages are twofold – the first in terms of reduced time queries to the closest plane vis-a-vis the closest voxel to current quadrotor pose, the second in terms of better navigation performance evaluated based on safety, and trajectory length.

This chapter is organized as follows. The pipeline is described in section 5.1. We then report the results of our experiments and validation in section 5.4 and conclude in section 5.5.

5.1 Pipeline Overview

5.1.1 Overview

Fig.5.1 presents the overview of our perception and control pipeline that enables quadrotors equipped with a single camera and an IMU to navigate safely amongst skyscrapers.

Perception Module: Our perception module's primary role is to represent the obstacles in the environment (skyscrapers) as axis-aligned cuboids. We perform this in the following manner. The RGB images captured from a monocular camera are pre-processed to extract key points. Monocular Visual-Inertial Odometry (VIO) [39] esti- mates upto-scale poses from these key points. Then, metric poses are computed by fusing the upto-scale poses with pre-integrated IMU measurements. VIO also optimizes the computed poses using factor- graph and computes a sparse 3D map. We use Plane-RCNN [31], a data driven network to detect plane segment masks. The sparse 3D map of triangulated points is clustered using the plane masks. Each triangulated 3D key-point in the point cloud is then associated with a plane if the corresponding 2D key-point observation falls within a plane segment mask in the image. We find the plane parameters of the observed plane segmented using RANSAC (RANdom SAmple Consensus) on each cluster.

Planning Module: The representation of the world in terms of cuboids allows us to perform fast distance computation of any query-point with respect to these cuboids. Our optimizer draws several random trajectories and computes the collision-avoidance estimates along each of these to compute the safest trajectory. We then use sequential convex programming with half-space constraints induced by cuboid planes to further refine the safest trajectory.

5.2 Monocular VINS based Cuboid Reconstruction

Assumptions

We assume that the buildings are planar and feature rich. We further, expect a reliable state-estimate from the VINS system, but we acknowledge the uncertainty in the estimated 3D planes obtained from

the triangulated point clouds of the monocular VINS. We also assume that the VINS estimate of gravity axis is accurate enough.

5.2.1 Monocular VINS

Although a detailed description of VINS-Mono [39] is beyond the scope of this chapter, we briefly describe the steps involved in estimating the odometry from RGB images and IMU measurements.

The system starts with measurement pre-processing, wherein 2D key points are extracted and tracked. Simultaneously, the IMU measurements between two consecutive frames are pre-integrated. Then initialization procedure provides all necessary values, including pose, velocity, gravity vector, gyroscope bias, and three-dimensional (3D) key-points, for bootstrapping the subsequent sliding-window-based nonlinear optimization-based VIO. We use the resulting camera pose to transfer the 3D key points into the world frame that are subsequently used in step 5.2.3 for plane reconstruction.

5.2.2 Data driven 2D plane segmentation

We use PlaneRCNN [31], a data driven network, to generate the 2D plane segment masks from the RGB image. It consists of three modules viz., a plane detection network, segmentation refinement network, and a warping loss module. In order to make the plane segmentation process faster, we propagate the detected plane segment mask to the succeeding four successive frames using dense optical flow as described in [11], while using PlaneRCNN for every 1 out 5 RGB image frames. The obtained plane masks are used for clustering the 3D key points in the following section.

5.2.3 3D plane reconstruction

Each 3D key point triangulated by VINS is then associated with a plane if the corresponding 2D key point observation falls within a plane segment mask in the image. This results in the clustering of 3D points. For each cluster, the parameters $(\mathbf{n}, d)$ representing the plane normal and the plane offset respectively are estimated using RANSAC. This serves as the observation of the 3D plane corresponding to the 2D plane mask detected in the RGB image.

5.3 Trajectory Planning

5.3.1 SCP-MMD planner

Our planner generates multiple point-to-point trajectories using the spatial covariance matrix proposed in [23]. Each trajectory is ranked based on the sum of collision costs of its waypoints. Then the trajectory with least cost is chosen for further optimization with smoothness and plane half-space constraints.

Plane half-space constraint: For a given waypoint $\mathbf{x}_{ij}$, the analytical expression to compute the distance to the plane with parameters $(\mathbf{n}_k, \mathbf{d}_k)$ is given by

$$\mathbf{n}_k^T.\mathbf{x}_{ij} + d_k \tag{5.1}$$

Based on the sign of the value of the constraint (5.1), each plane divides a 3D space into two halfs viz., positive and negative half-spaces. The normals of the planes are defined such that they point towards the camera origin. So, the space outside a cuboid is the union of positive half-spaces of all 6 planes defined by the cuboid's faces. Similarly, the space inside the cuboid is an intersection of negative half-spaces of all 6 planes defined by the cuboid faces. So, to avoid collisions, the waypoints should be in the positive half-space of all cuboids in the scene.

We make a simplifying assumption that the obstacle size and orientation are deterministic, and we account for the uncertainty by enlarging the size of the obstacle. The basic optimization problem can be described in the following manner.

$$\sum_t \ddot{x}(t)^2 + \ddot{y}(t)^2 + \ddot{z}(t)^2 \tag{5.2a}$$

$$(x(t_0), y(t_0), z(t_0), \dot{x}(t_0), \dot{y}(t_0), \dot{z}(t_0), \ddot{x}(t_0), \ddot{y}(t_0), \ddot{z}(t_0)) = \mathbf{b}_0. \tag{5.2b}$$

$$(x(t_f), y(t_f), z(t_f), \dot{x}(t_f), \dot{y}(t_f), \dot{z}(t_f), \ddot{x}(t_f), \ddot{y}(t_f), \ddot{z}(t_f)) = \mathbf{b}_f. \tag{5.2c}$$

$$-v_{max} \leq \{\dot{x}(t), \dot{y}(t), \dot{z}(t)\} \leq v_{max}, \tag{5.2d}$$

$$-a_{max} \leq \{\ddot{x}(t), \ddot{y}(t), \ddot{z}(t)\} \leq a_{max} \tag{5.2e}$$

$$d_c(x(t), y(t), z(t)) \geq 0, \forall t \tag{5.2f}$$

The cost function (5.2a) minimizes the norm of the acceleration input over the planning horizon. The equality constraints (5.2b)-(5.2c) enforces the boundary conditions on the trajectory. The inequalities (5.2d)-(5.2e) are the bounds on the velocities and accelerations. The last set of constraints (5.2f) ensure that the signed-distance at the query point $x(t), y(t), z(t)$ with respect to the cuboid obstacle c is greater than zero. To solve (5.2a)-(5.2f), we adopt a way-point parametrization of the trajectory and approximate the derivatives through finite differences.

Initial Guess: Optimization (5.2a)-(5.2f) is non-convex due to the presence of (5.2f). Thus, we need to provide an initial guess for its solution that will be used to kick-start an SCP-based optimizer. Typically, initialization trajectory is computed through sampling based planners. However, we take a fundamentally different approach. We draw a set of random trajectories from a Gaussian distribution centered around the straight line connecting the start and goal location and covariance given by [23]. We then compute the collision cost l_{dist} over all these trajectories and choose one with the lowest value. This lowest cost trajectory will be used as the initial guess for solving (5.2a)-(5.2f).

5.4 Experiments and Validation

In this section, we demonstrate the ability of our pipeline, *UrbanFly* to safely navigate in an environment populated with skyscrapers. We further argue that the voxel grid representation can be sub-optimal in a monocular setting due to inherent noise in monocular reconstruction and sparsity of the point clouds. Moreover, we qualitatively and quantitatively prove the efficiency of *UrbanFly* to generate smooth trajectories for long distances and traverse it in a way that the SLAM does not break. We also perform ablation studies to qualitatively prove that our approach tends to generate collision-free trajectories.

5.4.1 Simulation Setup

We used Airsim [46], a highly integrated and complete simulation framework for multirotor. It implements a physics engine, flight controller, and photorealistic scene. Airsim runs inside Unreal Engine [10] Editor. Airsim drone has a range of sensors that can be used, but we equip the drone with an RGB camera, ahead, and an IMU sensor. We test our algorithms in two environments explained in the next subsection. We run our planners on a computer with 16GB RAM, core i7-10710 CPU. We used a dedicated computer to run Airsim simulations and Unreal Engine. It is equipped with NVIDIA-1070 graphics card, AMD Ryzen 7 3800x 8-core processor × 16 CPU, and 64GB RAM.The mapping module and CEM planner are programmed in C++. The SCP-MMD planner was implemented in Python using CVXPY[9].

As discussed in 5.2.2, on this setup PlaneRCNN provides plane segmentation at a rate of 2-3Hz for images with resolution of 640×480. Therefore, we increase its frequency to 10Hz by using dense optical flow [11].

5.4.2 Simulation Environments

We test our approach on the following two data-sets.

5.4.2.1 SquareStreet (Synthetic Dataset)

Thanks to Unreal Engine (UE) Editor [10], we were able to create custom scenes to test our approach. We created a street in the shape of a square, having 47 buildings spread over 0.16 square kilometers. The environment is feature-rich and, therefore, can be easily used for benchmarking monocular vision-based

(a) Square Street (b) Yingrenshi from UrbanScene3D

Figure 5.2: Simulation environments

perception and planning pipelines. The whole environment is modular and can be easily expanded or configured to meet custom requirements.

5.4.2.2 Yingrenshi (Real city model)

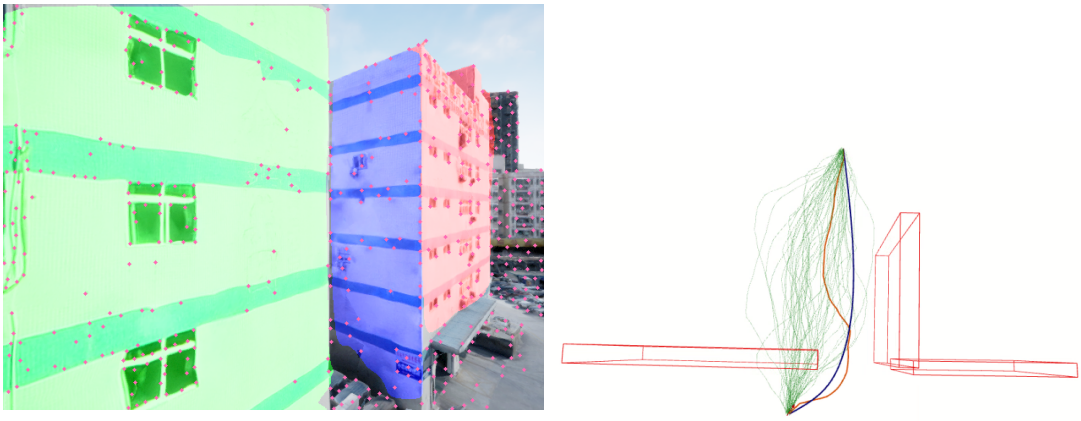
UrbanScene3D [33] is a dataset with real and virtual city models which can be imported into Unreal Engine. Yingrenshi, shown in 5.3. It has 252 buildings spread over 1sq. Km of area. This model is a replica of real-world buildings and streets. This environment helps us to evaluate our approach in a real-world setting.

5.4.2.3 Qualitative Analysis

The qualitative demonstration of *UrbanFly* in both environments is depicted in Fig 5.1. We can observe that *UrbanFly* was able to map the environments and plan long range trajectories successfully.

5.5 Conclusions and Future Work

In this chapter, we demonstrated a navigation on monocular SLAM maps of urban high-rise environments for the first time. We showed how constructing a geometrical representation of the world can significantly counter the inherent sparsity of the point clouds generated by the VINS system. We developed a trajectory optimizer that can leverage our chosen world model. In particular, fast distance queries allowed by our cuboid representation of the obstacles proved key in deriving tractable collision estimates and minimizing it along with conventional smoothness costs. Our optimizer showed improvement over several strong baselines in terms of success-rate, trajectory smoothness, arc-length, etc.



(a) Segmented Facades (in colour) and VINS features (red dots)

(b) Estimated cuboids and planned trajectories

Figure 5.3: Qualitative results of the proposed pipeline

Our work has robust real-world application in air logistics, search and rescue, structural inspection, etc. Thus, our future endeavors are geared towards successfully deploying our approach on actual hardware.

Chapter 6

Conclusions and Future Work

Structure estimated using the proposed approach can be extended to higher level features like fitting planes for facade reconstruction of buildings, etc. for future work. The motion estimation if extended can be used in a SLAM system. To operate in the metric scale, integrating it within a visual inertial system can be tried.

Our work has robust real-world application in air logistics, search and rescue, structural inspection, etc. Thus, our future endeavors are geared towards successfully deploying our approach on actual hardware. To this end, we also aim to expand our approach to not only include perception but also quadrotor's state uncertainty. Furthermore, we would like to integrate the advantages accrued due to structures present in the scene and their uncertainty characterizations to existing voxel based representations.

Related Publications

[1] (*Oral - Full Paper*) Ayyappa Swamy Thatavarthy, Tanu Sharma, Harsh Sankhla, Mukul Khanna and K. Madhava Krishna. “Multi-view Planarity Constraints for Skyline Estimation from UAV Images in City Scale Urban Environments.” VISIGRAPP (2021).

[2] Ayyappa Swamy Thatavarthy, Tanu Sharma and K. Madhava Krishna, ”A new geometric approach for three view line reconstruction and motion estimation in Manhattan Scenes,” 2021 18th Conference on Robots and Vision (CRV), 2021, pp. 135-141, doi: 10.1109/CRV52889.2021.00026.

[3] (*Submitted to IROS 2022*) Ayyappa Swamy Thatavarthy, Sudarshan S. Haritas, Arun Kumar Singh, K. Madhava Krishna. ”UrbanFly: Uncertainty-Aware Planning for Navigation Amongst High-Rises with Monocular Visual-Inertial SLAM Maps”.

Other Publications

[1] M. Khanna, T. Sharma, A. S. Thatavarthy and K. M. Krishna, ”Building Facades to Normal Maps: Adversarial Learning from Single View Images,” 2021 18th Conference on Robots and Vision (CRV), 2021, pp. 33-40, doi: 10.1109/CRV52889.2021.00009.

Bibliography

[1] P. Agarwal, G. D. Tipaldi, L. Spinello, C. Stachniss, and W. Burgard. Robust map optimization using dynamic covariance scaling. In *2013 IEEE International Conference on Robotics and Automation*, pages 62–69, 2013.

[2] S. Agarwal, K. Mierle, and Others. Ceres solver. `http://ceres-solver.org`.

[3] Z. Akbulut, S. Özdemir, H. Acar, and F. Karsli. Automatic building extraction from image and lidar data with active contour segmentation. *Journal of the Indian Society of Remote Sensing*, 46(12):2057–2068, Dec. 2018.

[4] I. Alzugaray, L. Teixeira, and M. Chli. Short-term uav path-planning with monocular-inertial slam in the loop. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2739–2746, 2017.

[5] A. Bartoli and P. Sturm. Structure-From-Motion Using Lines: Representation, Triangulation and Bundle Adjustment. *Computer Vision and Image Understanding*, 100:416–441, 2005.

[6] L. Blackmore, M. Ono, A. Bektassov, and B. C. Williams. A probabilistic particle-control approximation of chance-constrained stochastic predictive control. *IEEE Transactions on Robotics*, 26(3):502–517, 2010.

[7] G. Costante, J. Delmerico, M. Werlberger, P. Valigi, and D. Scaramuzza. *Exploiting Photometric Information for Planning Under Uncertainty*, pages 107–124. 01 2018.

[8] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes, 2017.

[9] S. Diamond and S. Boyd. CVXPY: A Python-embedded modeling language for convex optimization. *Journal of Machine Learning Research*, 17(83):1–5, 2016.

[10] Epic Games. Unreal engine.

[11] G. Farnebäck. Two-frame motion estimation based on polynomial expansion. In J. Bigun and T. Gustavsson, editors, *Image Analysis*, pages 363–370, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg.

[12] R. Gomez-Ojeda, F. Moreno, D. Zuñiga-Noël, D. Scaramuzza, and J. Gonzalez-Jimenez. Pl-slam: A stereo slam system through the combination of points and line segments. *IEEE Transactions on Robotics*, 35(3):734–746, 2019.

[13] B. Gopalakrishnan, A. K. Singh, and K. Krishna. Closed form characterization of collision free velocities and confidence bounds for non-holonomic robots in uncertain dynamic environments. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4961–4968, 2015.

[14] W. N. Greene, K. Ok, P. Lommel, and N. Roy. Multi-level mapping: Real-time dense monocular slam. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 833–840, 2016.

[15] R. Grompone von Gioi, J. Jakubowicz, J.-M. Morel, and G. Randall. LSD: a Line Segment Detector. *Image Processing On Line*, 2:35–55, Mar. 2012.

[16] G. Guennebaud, B. Jacob, et al. Eigen v3. http://eigen.tuxfamily.org, 2010.

[17] S. S. Harithas, R. D. Yadav, D. Singh, A. K. Singh, and K. M. Krishna. Cco-voxel: Chance constrained optimization over uncertain voxel-grid representation for safe trajectory planning, 2021.

[18] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition, 2004.

[19] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn, 2017.

[20] M. Hofer, M. Maurer, and H. Bischof. Line3d: Efficient 3d scene abstraction for the built environment. 10 2015.

[21] S. Huang, N. Kwok, G. Dissanayake, Q. Ha, and G. Fang. Multi-step look-ahead trajectory planning in slam: Possibility and necessity. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pages 1091–1096, 2005.

[22] K. Joo, T. Oh, J. Kim, and I. S. Kweon. Robust and globally optimal manhattan frame estimation in near real time. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(3):682–696, 2019.

[23] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, and S. Schaal. Stomp: Stochastic trajectory optimization for motion planning. In *2011 IEEE International Conference on Robotics and Automation*, pages 4569–4574, 2011.

[24] D. Khurana, S. Sankhla, A. Shukla, R. Varshney, P. Kalra, and S. Banerjee. A grammar-based gui for single view reconstruction. In *Proceedings of the Eighth Indian Conference on Computer Vision, Graphics and Image Processing*, ICVGIP '12, New York, NY, USA, 2012. Association for Computing Machinery.

[25] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard. G2o: A general framework for graph optimization. In *2011 IEEE International Conference on Robotics and Automation*, pages 3607–3613, 2011.

[26] C. Leung, S. Huang, and G. Dissanayake. Active slam using model predictive control and attractor based exploration. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5031, 2006.

[27] J. Lezama, G. Randall, and R. Grompone von Gioi. Vanishing Point Detection in Urban Scenes Using Point Alignments. *Image Processing On Line*, 7:131–164, July 2017.

[28] H. Li, Y. Xing, J. Zhao, J. Bazin, Z. Liu, and Y. Liu. Leveraging structural regularity of atlanta world for monocular slam. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 2412–2418, 2019.

[29] H. Li, J. Yao, J. Bazin, X. Lu, Y. Xing, and K. Liu. A monocular slam system leveraging structural regularity in manhattan world. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2518–2525, 2018.

[30] C. Liu, K. Kim, J. Gu, Y. Furukawa, and J. Kautz. Planercnn: 3d plane detection and reconstruction from a single image, 2018.

[31] C. Liu, K. Kim, J. Gu, Y. Furukawa, and J. Kautz. Planercnn: 3d plane detection and reconstruction from a single image. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4445–4454, 2019.

[32] C. Liu, J. Yang, D. Ceylan, E. Yumer, and Y. Furukawa. Planenet: Piece-wise planar reconstruction from a single rgb image, 2018.

[33] Y. Liu, F. Xue, and H. Huang. Urbanscene3d: A large scale urban scene dataset and simulator. 2021.

[34] X. Lu, J. Yao, H. Li, Y. Liu, and X. Zhang. 2-line exhaustive searching for real-time vanishing point estimation in manhattan world. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*, March 2017.

[35] A. L. Majdik, C. Till, and D. Scaramuzza. The Zurich urban micro aerial vehicle dataset. *The International Journal of Robotics Research*, 36(3):269–273, Mar. 2017.

[36] A. L. Majdik, C. Till, and D. Scaramuzza. The zurich urban micro aerial vehicle dataset. *The International Journal of Robotics Research*, 36(3):269–273, 2017.

[37] C. Mostegel, A. Wendel, and H. Bischof. Active monocular localization: Towards autonomous monocular exploration for multirotor mavs. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3848–3855, 2014.

[38] S. N. J. Poonganam, B. Gopalakrishnan, V. S. S. B. K. Avula, A. K. Singh, K. M. Krishna, and D. Manocha. Reactive navigation under non-parametric uncertainty through hilbert space embedding of probabilistic velocity obstacles. *IEEE Robotics and Automation Letters*, 5(2):2690–2697, 2020.

[39] T. Qin, P. Li, and S. Shen. Vins-mono: A robust and versatile monocular visual-inertial state estimator. *IEEE Transactions on Robotics*, 34(4):1004–1020, 2018.

[40] S. Ramalingam and M. Brand. Lifting 3d manhattan lines from a single image. In *2013 IEEE International Conference on Computer Vision*, pages 497–504, 2013.

[41] S. Ranade and S. Ramalingam. Novel single view constraints for manhattan 3d line reconstruction. In *2018 International Conference on 3D Vision (3DV)*, pages 625–633, 2018.

[42] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation, 2015.

[43] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. M. Lopez. The SYNTHIA Dataset: A Large Collection of Synthetic Images for Semantic Segmentation of Urban Scenes. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3234–3243, Las Vegas, NV, USA, June 2016. IEEE.

[44] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. M. Lopez. The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[45] G. Schindler and F. Dellaert. Atlanta world: an expectation maximization framework for simultaneous low-level edge grouping and camera calibration in complex man-made environments. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, volume 1, pages I–I, 2004.

[46] S. Shah, D. Dey, C. Lovett, and A. Kapoor. Airsim: High-fidelity visual and physical simulation for autonomous vehicles. In *Field and Service Robotics*, 2017.

[47] J. Straub, O. Freifeld, G. Rosman, J. J. Leonard, and J. W. Fisher. The manhattan frame model—manhattan world inference in the space of surface normals. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(1):235–249, 2018.

[48] X. Wang, R. Girshick, A. Gupta, and K. He. Non-local neural networks, 2017.

[49] T. Werner and A. Zisserman. New techniques for automated architecture reconstruction from photographs. In *European Conference on Computer Vision*, volume 2, pages 541–555. Springer-Verlag, 2002.

[50] F. Yang and Z. Zhou. Recovering 3d planes from a single image via convolutional neural networks, 2018.

[51] G. Zhang, J. H. Lee, J. Lim, and I. H. Suh. Building a 3-d line-based map using stereo slam. *IEEE Transactions on Robotics*, 31(6):1364–1377, 2015.

[52] H. Zhou, K. Peng, D. Zhou, W. Fan, and Y. Liu. Uncertainty analysis of 3d line reconstruction in a new minimal spatial line representation. *Applied Sciences*, 10(3):1096, Feb 2020.

[53] Q.-Y. Zhou, J. Park, and V. Koltun. Open3D: A modern library for 3D data processing. *arXiv:1801.09847*, 2018.

[54] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe. Unsupervised learning of depth and ego-motion from video. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6612–6619, 2017.

[55] Y. Zhou, H. Qi, Y. Zhai, Q. Sun, Z. Chen, L.-Y. Wei, and Y. Ma. Learning to Reconstruct 3D Manhattan Wireframes from a Single Image. *arXiv:1905.07482 [cs]*, May 2019. arXiv: 1905.07482.

[56] H. Zhu and J. Alonso-Mora. Chance-constrained collision avoidance for mavs in dynamic environments. *IEEE Robotics and Automation Letters*, 4(2):776–783, 2019.

[57] D. Zou, Y. Wu, L. Pei, H. Ling, and W. Yu. Structvio: Visual-inertial odometry with structural regularity of man-made environments. *IEEE Transactions on Robotics*, 35(4):999–1013, 2019.