Abstract

For safe autonomous/semi-autonomous Unmanned Aerial Vehicles (UAV) flights navigation systems need to capture and fuse information from all onboard sensors including visual information captured by cameras. Raw images need to be processed and more structured information extracted for navigation planning. One of the image processing tasks is 3D scene reconstruction, when depth information is recovered from plain images captured by a flying camera. This operation is called Depth Estimation. State of the art literature body for depth estimation for monocular camera images consists of a wide range of works covering approaches from classic computer vision to deep learning data driven methods. The majority of works on monocular depth estimation are concerned either with indoor or outdoor images captured from ground.

This work aims to research classic computer vision and deep learning methods for depth estimation with application to UAV localization.

Contents

Declaration of Authorship i

Abstract ii

Acknowledgements iii

Contents iv

List of Figures vi

List of Tables vii

Abbreviations viii

1 Introduction 1

2 Literature review 3
- 2.1 Background . . . 3
  - 2.1.1 UAV navigation . . . 3
  - 2.1.2 Simultaneous Localization and Mapping (SLAM) and navigation . 5
  - 2.1.3 Kalman filter . . . 7
  - 2.1.4 Computer vision . . . 9
    - 2.1.4.1 Image formation and camera extrinsic and intrinsic . . . 9
    - 2.1.4.2 Multiple images of the same scene from different places . 11
- 2.2 Depth estimation in stereo camera setup . . . 13
- 2.3 Interest points finding with Scale Invariant Feature Transform (SIFT) algorithm . . . 14
- 2.4 Depth estimation with Semi-global matching . . . 16
- 2.5 Monocular Depth Estimation using Deep Learning . . . 17
  - 2.5.1 Supervised deep learning approach . . . 17
  - 2.5.2 Unsupervised Deep Learning approach . . . 19
  - 2.5.3 Semi-supervised Deep Learning . . . 22
- 2.6 Critical analysis . . . 24

3 Methodology 25
- 3.1 Overview . . . 25
- 3.2 Project phase details . . . 26

3.2.1 Phase 1: Depth estimation from Interest points 26
3.2.2 Phase 2: Pixel-wise estimation with Deep Learning model from a single image 26
3.2.3 Phase 3: Sequence-to-sequence DE DL model 27
3.2.4 Phase 4: Application of Kalman filter to enhance DE of a DL model 28
3.3 Evaluation strategy 28
3.3.1 Metrics 29
3.4 Requirements analysis 29
3.4.1 Functional requirements 30
3.4.2 Non-functional requirements 30

4 Project management 31
4.1 Work plan 31
4.2 Risks analysis 33
4.3 Professional, Legal, Ethical and Social Issues 34
4.3.1 Professional Issues 34
4.3.2 Legal Issues 34
4.3.3 Ethical and Social Issues 34

Bibliography 36

List of Figures

2.1 Vision-aided sensor fusion architecture 5
2.2 UAV navigation system 6
2.3 Pinhole camera model 10
2.4 3D to 2D projection 10
2.5 Two-view analysis. 12
2.6 Stereo camera setup for depth estimation 13
2.7 Finding orientation of a blob in SIFT detector 15
2.8 SIFT descriptor forming procedure 16
2.9 Model architecture by Eigen et al. 17
2.10 DeMoN model architecture for predicting depth map and ego-motion . . . 19
2.11 DANet network architecture 19
2.12 Convolutional LSTM encoder and decoder of DepthNet 20
2.13 Auto-encoder setup for self-supervised learning 20
2.14 Pipeline for self-supervised learning of dept and pose 21
2.15 Depth net with GCN 22
2.16 Sequence-to-sequence depth estimator with ConvGRU blocks 22
2.17 Semi-supervised loss function for MDE model by Kuznietsov et al. 23
2.18 Training framework for DE from image and sparse LIDAR measurements 23
2.19 AdaDepth encoder-decoder architecture 24

4.1 Project schedule. Part 1. 31
4.2 Project schedule. Part 2. 32
4.3 Project schedule. Part 3. 32
4.4 Project schedule. Part 4. 33

List of Tables

4.1 Risk analysis 33

Abbreviations

UAV	**U**nmanned **A**erial **V**ehicles
DE	**D**epth **E**stimaion
SLAM	**S**imultaneous **L**ocalization **a**nd **M**apping
GNSS	**G**lobal **N**avigation **S**atellite **S**ystem
GPS	**G**lobal **P**ositioning **S**ystem
IMU	**I**nertial **M**easurement **U**nit
SIFT	**S**cale **I**nvariant **F**eature **T**ransform
DL	**D**eep **L**earning
GT	**G**round **T**ruth
MDE	**M**onocular **D**epth **E**stimation
FLOPs	**F**loating **P**oint **O**peration**s**
LSTM	**L**ong **S**hort **T**erm **M**emory
GCN	**G**raph **C**onvolutional **N**etwork
CNN	**C**onvolutional **N**eural **N**etwork
DM	**D**epth **M**ap

Chapter 1

Introduction

The use of Unmanned Aerial Vehicles (UAV) can be beneficial in numerous activities, e.g. delivery, high-altitude surveillance of buildings and constructions, disaster recovery, and military operations. The navigation process of a UAV is not a simple task and has several challenges:

1. Sole reliance on Global Navigation Satellite System (GNSS) signal can be not desirable, as the signal can be not reliable (especially in urban landscape) or it can be spoofed or just disabled.
2. If navigation is done through remote control, then a UAV can experience a communication loss because of natural or intentional signal interference.
3. Safe operation of a UAV mandates reliable Obstacle Avoidance to all objects static (buildings, trees, etc.) and dynamic such as other aircraft. This problem becomes more important if we consider the possibility of sudden weather changes, e.g. strong unpredictable wind gusts.

To build a navigation system which does not rely on GNSS different sensors are used. The most basic and not expensive sensors for navigation are Inertial Measurement Unit (IMU) and camera. With help of visual information, it is possible to achieve the following tasks:

- Locate the current position of UAV and create a map of the environment

- Keep relative orientation
- Implement a Visual compass and Visual Odometer
- Make landing more accurate and safe
- Avoid obstacles

As a UAV operates in 3D space its navigation system should be able to sense all three dimensions of the environment. 3D scene reconstruction with a stereo camera setup for UAV navigation faces hard limitations as the maximum distance between cameras on a UAV is relatively small which limits the distance for reliable 3D scene sensing. Hence the main source of visual information for 3D scene reconstruction is a sequence of images captured by an onboard camera. In computer vision, the process of 3D scene reconstruction from a single camera is called Monocular Depth Estimation (MDE).

The aim of this research project is to explore different approaches for MDE applied to UAV navigation without a need to communicate with external systems during flight time. In particular, we will investigate:

- Interest points based method for depth estimation with the use of Kalman filter
- Pixel-wise MDE with use of Deep Learning

The performance of both approaches will be evaluated on data generated from a simulator and data provided by Technology Innovation Institute (Abu Dhabi), additionally, the feasibility of running on an onboard CPU will be estimated.

The rest of the document is organized as follows:

- **Chapter 2:** Provides the background of UAV navigation, computer vision, and reviews the literature for classical and Deep Learning based MDE methods.
- **Chapter 3:** Presents the methodology of project implementation, evaluation metrics and requirements.
- **Chapter 4:** Presents the project plan, risk analysis, and discusses Professional, Legal, Ethical and Social aspects of the project.

Chapter 2

Literature review

2.1 Background

2.1.1 UAV navigation

UAVs have a wide range of applications. Newer applications require putting UAV in a complex landscape where manual control by a human operator cannot be enough due to the dynamism of the environment. At the same time, there is a demand for autonomous UAVs in traditional applications such as delivery, emergency handling, and surveying. All these factors rise interest in researching of autonomous and semi-autonomous navigation systems.

Navigation capability is wide and includes the following: perception, localization, motion planning and motion control [Gyagenda et al., 2022].

For the perception of the environment, a range of sensor types is being used:

1. Inertial Measurement Unit (IMU). It is the most standard and basic type of sensor used in UAVs. It includes three gyroscopes and three accelerometers
2. Visual sensors: monocular, stereo, depth, thermal cameras
3. LiDAR and ultrasonic rangefinders
4. Radars

5. GNSS receiver

The modern requirements push UAVs to be lightweight to enable caring of more load, energy effective for improved reaching range, and cost-effective. Hence, the most common sensor set for an off-the-shelf UAV includes: IMU, monocular camera and GNSS receiver.

The localization capability of UAV is estimating of its position. The most basic technique is the reading of GNSS information which is vulnerable to spoofing, jamming, unreliable in difficult landscapes (especially urban), and inaccessible indoors. Another basic technique relies on using IMU data and control command history essentially enabling dead reckoning. The main drawback of the last solution is a significant drift of position with time.

The most information-reach sensor is the onboard camera. The main challenge of using camera data in a navigation system is the need to process the raw images and extract the information which can be used for navigation and localization. Another challenge for visual-based sensing is the variation in the landscape, illumination and presence of dynamic objects.

The pioneering research in the use of image information for altitude estimation was done by Cherian et al. [2009] The authors developed a set of image features and a machine learning algorithm for regression of the single number altitude from a high altitude image. The issues of the proposed method are low precision at low altitudes and images with weak texture.

Another direction of research of visually aided source of navigation data is Visual Odometer. By applying photogrammetry methods to a sequence of images taken from a flying UAV it is possible to measure a relative change in the pose to use for navigation. Despite the long history of photogrammetry, new methods are being proposed. For example, a contemporary paper by Liu et al. [2022] presents a method for a reliable high-altitude orientation estimation by exploiting clusters of key points.

As a UAV has a range of sensors with different noise levels and different reliability of the measured information, it requires a navigation system to be able to fuse the data for predictable localization with minimum drift. Conte and Doherty [2008] proposed a navigation system which can fuse different sources of data. The proposed system exploits the physical movement constraints of a UAV by applying 12-state Kalman filter for the UAV state estimation. The proposed system is shown in Figure 2.1

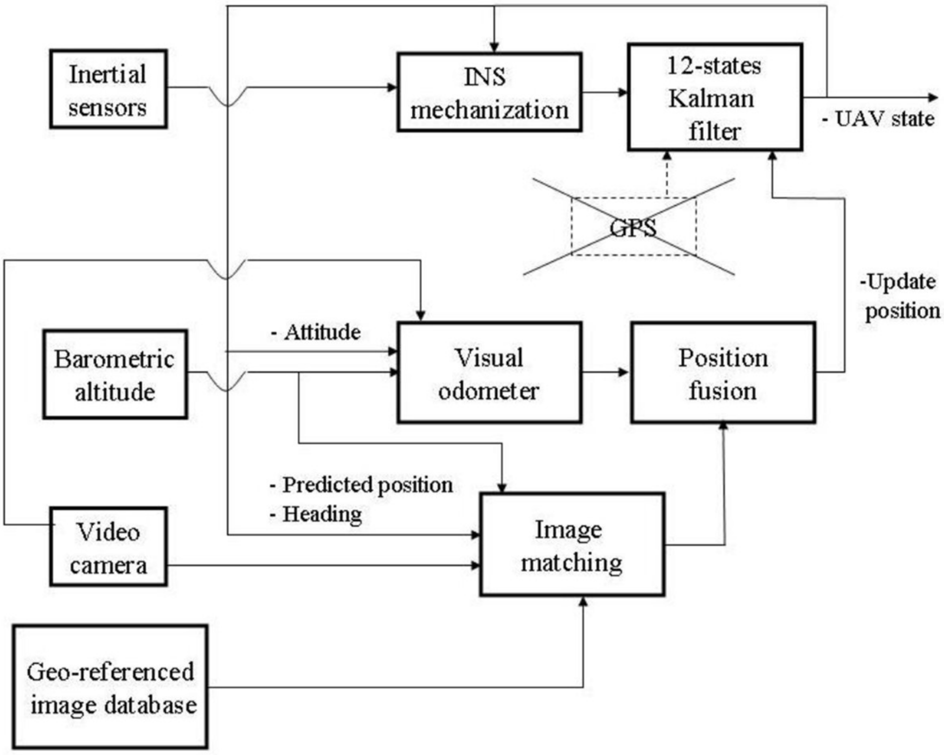


Figure 2.1: Vision-aided sensor fusion architecture. [Conte and Doherty, 2008]

The main advancement of the system proposed in [Conte and Doherty, 2008] is the Image Matching Block for estimating the absolute position of UAV by matching images captured by the onboard camera with images from the geo-referenced database. When Image Matching Block finds a reliable match of the current view with an image from the database the estimated geo-referenced location is used for updating the Kalman filter. The system demonstrated low drift with a disabled GPS receiver and was able to run the UAV along the test route. The main drawback of the system is the requirement for geo-referenced images, which are susceptible to out-dating and labour-intensive to collect.

2.1.2 Simultaneous Localization and Mapping (SLAM) and navigation

To fully exploit the agility of mobile robots in a dynamic environment, especially in UAVs, there is a demand for algorithms for automatic path planning and collision avoidance. It is very costly to build a full map of the 3D environment in which the UAV is supposed to operate.

To tackle the problem of unavailable full mapping of the environment, SLAM technique was proposed for building and/or updating the map and determining the current position of moving robot in this environment at the same time [1].

SLAM starts with gathering measurements from all onboard sensors, then keeping data collection SLAM algorithm does [Gupta and Fernando, 2022]:

1. Recording of current observation
2. Detection of predefined landmarks
3. Creation of new landmarks
4. UAV trajectory tracking
5. UAV pose estimation

A drawback of SLAM-produced maps is their sparsity which leads to problems with path planning and collision avoidance in realistic autonomous tasks [Teixeira et al., 2018]. Teixeira et al. [2018] proposed a system which along with SLAM-produced information uses dense 3D maps for path planning. The proposed system is presented in Figure 2.2.

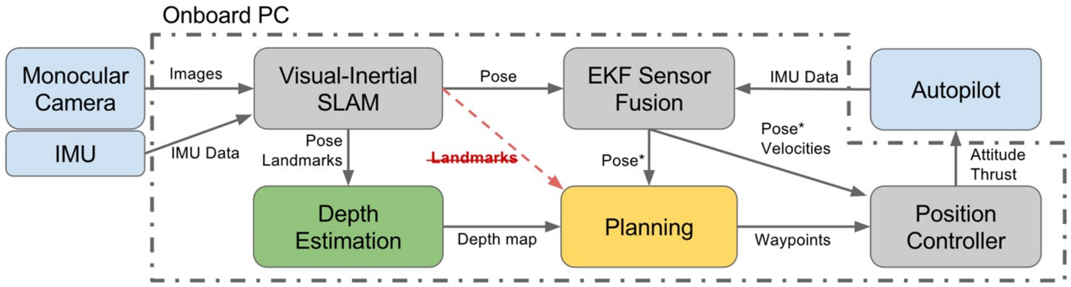


FIGURE 2.2: UAV navigation system. [Teixeira et al., 2018]

The important block in the proposed system is Depth Estimator (DE). The task of DE is to provide a 3D reconstruction of the scene perceived by available onboard sensors. The sensor set can be any combination of camera, LiDAR, ultrasonic sensor, radar, and RGB-D camera. This enhancement enables better path planning in a dynamic environment.

[1] https://en.wikipedia.org/wiki/Simultaneous_localization_and_mapping, Accessed: 2023-03-17

2.1.3 Kalman filter

Kalman filter is a mathematical tool which allows one to infer unmeasured variables from indirect and noisy data Grewal and Andrews [2015]. This tool is being used in many applications as well as in UAV navigation, especially in fusing data from numerous sensors, such as IMU, GPS, Visual compass and speedometer.

To simplify the presentation of the tool, let's consider the problem of tracking the true coordinates of a car on a 2D plane when we have x and y coordinates from a GNSS. In this system, we can identify two sources of error: measurement noise and system control noise. An example of the system control noise in UAVs can be a wind gust, a non-precise reaction of motors on control actions. To simplify the description more, let's consider that the car is moving at a constant speed.

To use a Kalman filter the following requirements need to be satisfied:

- A linear model of the observed dynamic system
- The measurement noise and system control error noise are uncorrelated

The variables for our problem are:

- Measurements: $z_i = \begin{pmatrix} z_i^x \\ z_i^y \end{pmatrix}$, where z_i^x and z_i^y are x and y coordinates received from a GNSS receiver
- Non-observed variables - system state: $x_i = \begin{pmatrix} x_i \\ y_i \\ s_i^x \\ s_i^y \end{pmatrix}$, where x_i and y_i are true x and y coordinates, s_i^x and s_i^y are speeds in x and y directions

The observations and the state of the system are related:

$$z_i = Hx_i + v_i \tag{2.1}$$

where H is a measurement matrix:

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \tag{2.2}$$

v_i is the measurement of Gaussian noise with the following covariance matrix:

$$R = \begin{bmatrix} \sigma^2 & 0 \\ 0 & \sigma^2 \end{bmatrix} \tag{2.3}$$

with σ variance in the measurement error.

The transition between states for the system is related by the state equation:

$$x_i = Ax_{i-1} + w_i \tag{2.4}$$

where A is the state transition matrix:

$$A = \begin{bmatrix} 1 & 0 & \delta t_i & 0 \\ 0 & 1 & 0 & \delta t_i \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.5}$$

w_i is a system error with the following covariance matrix:

$$Q = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma_s^2 & 0 \\ 0 & 0 & 0 & \sigma_s^2 \end{bmatrix} \tag{2.6}$$

where σ_s^2 is the speed variance.

To start Kalman filter algorithm we need:

- Initial state estimation x_0
- Initial estimation of the state error $P_0 = \begin{bmatrix} \sigma^2 & 0 & 0 & 0 \\ 0 & \sigma^2 & 0 & 0 \\ 0 & 0 & \sigma_s^2 & 0 \\ 0 & 0 & 0 & \sigma_s^2 \end{bmatrix}$

The Kalman filter algorithm is run discretely for every measurement cycle. It has the following steps:

1. Predict the state and the error covariance before correction:
   - $x_i^- = Ax_{i-1}$
   - $P_i^- = AP_{i-1}A^T + Q$
2. Calculate the Kalman gain: $K_i = P_i^- H^T \left(HP_i^- H^T + R\right)^{-1}$
3. Correct the estimate of the state and the error covariance:
   - $x_i = x_i^- + K_i \left(z_i - Hx_i^-\right)$
   - $P_i = \left(I - K_i H\right) P_i^-$

Conceptually, Kalman gain K_i is a measure of how reliable is the measurement; it serves as a coefficient to correct the prediction made by the dynamic model by the measurements at the current step.

2.1.4 Computer vision

2.1.4.1 Image formation and camera extrinsic and intrinsic

To understand the way how a camera forms a 2D image of 3D objects, we need to use a camera model. In this work, we look into a pinhole camera model (see Figure 2.3) where an image is formed on a film by rays passing through a pinhole from the outside world. Despite real modern cameras use lenses instead of a pinhole, this model allows to have simple equations which give quite close results to the lens camera model.

Let's introduce reference frames (Figure 2.4):

1. Real word coordinates – 3D.
2. Camera reference frame – 3D. The Centre of the frame is the pinhole itself – point O.
3. Image reference frame – 2D. Plane Π. The Centre of the frame is the centre of image c.

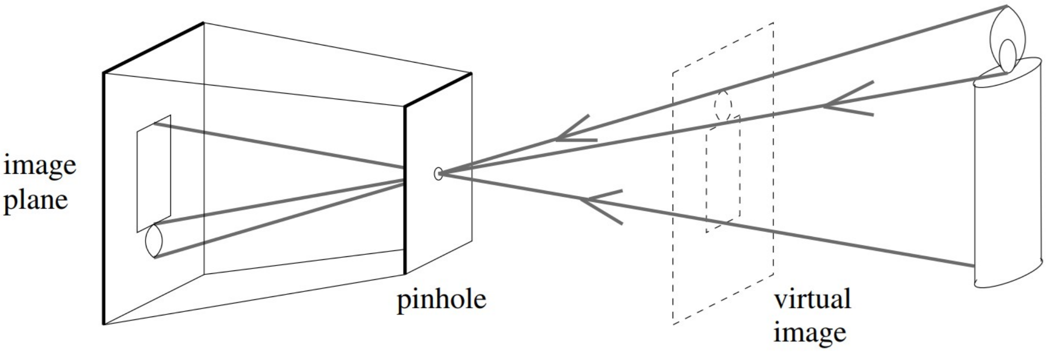


FIGURE 2.3: Pinhole camera model. [Forsyth et al., 2012]

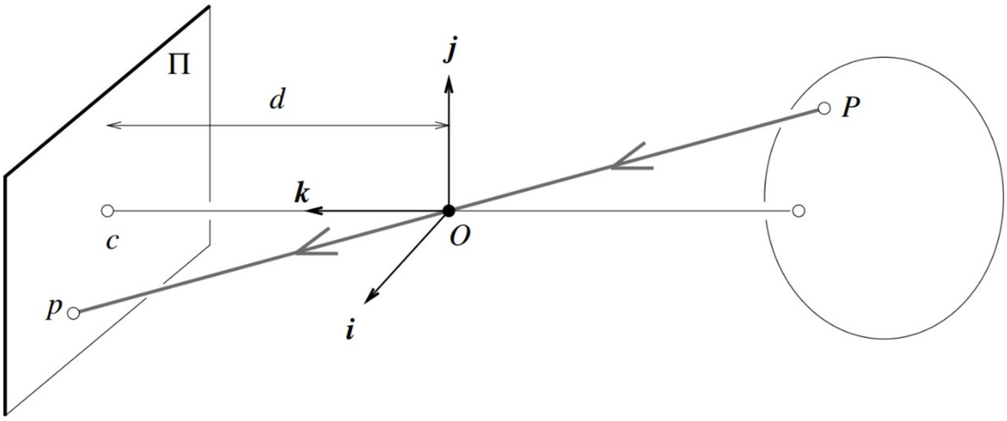


FIGURE 2.4: 3D world projection to 2D image. [Forsyth et al., 2012]

Translation of coordinates from the real world to the camera frame is made with the following equations:

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} = R \cdot T \cdot \begin{bmatrix} X_r \\ Y_r \\ Z_r \end{bmatrix} \tag{2.7}$$

In this equation:

- R is rotation matrix and is called as camera pose
- T is translation matrix
- X_r, Y_r, Z_r are coordinates of a point in the real world
- X_c, Y_c, Z_c are coordinates of the same real-world point in the camera coordinate system

The combined matrix $R \cdot T$ is called camera extrinsic. To convert coordinates from camera frame 3D to image frame 2D, we need to apply the following transformation:

$$\begin{bmatrix} X_i \\ Y_i \\ Z_c \end{bmatrix} = K \cdot \begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} \tag{2.8}$$

with X_i and Y_i are coordinates of the projection of the 3D world point on the image. Matrix K is called the camera intrinsic and has the following form:

$$K = \begin{bmatrix} f_x & 0 & x_0 \\ 0 & f_y & y_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.9}$$

Where:

- f_x and f_y are focal lengths – the distance between the pinhole and the sensor (image plane).
- x_0 and y_0 – offsets of the image centre from the perpendicular line to the image plane which crosses the pinhole.

In a true pinhole camera f_x and f_y are the same but in real cameras they are different.

Z_c value is called “depth” and represents the real value of the distance between the camera and the projected point in the 3D real world. The resulting image lacks the depth value as well as the pose of the camera (matrix R).

2.1.4.2 Multiple images of the same scene from different places

When we have several images of the same scene taken from different attitudes (see Figure 2.5), we can solve the following tasks:

1. Determine the relative orientation of the cameras
2. Reconstruct the depth of points seen on both images

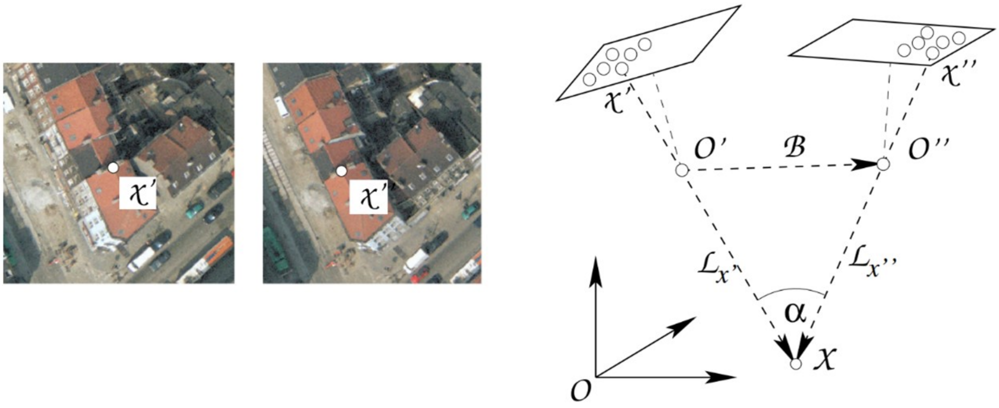


FIGURE 2.5: Two-view analysis. [Förstner and Wrobel, 2016]

It is worth mentioning, that the provided reconstruction does not give the real scale of the objects but preserves their shape.

In Figure 2.5 on the left we can see a point identified in both images of the same object taken from different attitudes. This point X is projected into two points X' and X'' on images (right side). Both cameras have their own poses and translation of camera 1 to camera 2 is done with vector B. Points $O^{'}$ and $O^{''}$ are pinholes of both cameras.

The projection lines $L_{x'}$ and $L_{x''}$ intersect in point X and thus lie in a plane. This fact allows us to write a co-planarity constraint [Förstner and Wrobel, 2016]:

$$X'^{T} \cdot M \cdot X'' = 0 \tag{2.10}$$

Where M is a 3×3 matrix.

According to [Forsyth et al., 2012] we can consider 2 cases:

1. Uncalibrated cameras:
   - M matrix is called Fundamental matrix.
   - It requires 5 points seen on both pictures to find a fundamental matrix.
2. Calibrated cameras:
   - M matrix is called Essential matrix.
   - It requires 3 points seen in both pictures to find a fundamental matrix.

In general, matrix M represents a photogrammetric model of a pair of images and allows us to solve both mentioned at the beginning of the paragraph tasks [Forsyth et al., 2012].

2.2 Depth estimation in stereo camera setup

If we have a setup of two calibrated cameras with a linear offset between them (see Figure 2.6), then we can calculate the coordinates (including depth) of a 3D world point which is captured on both images.

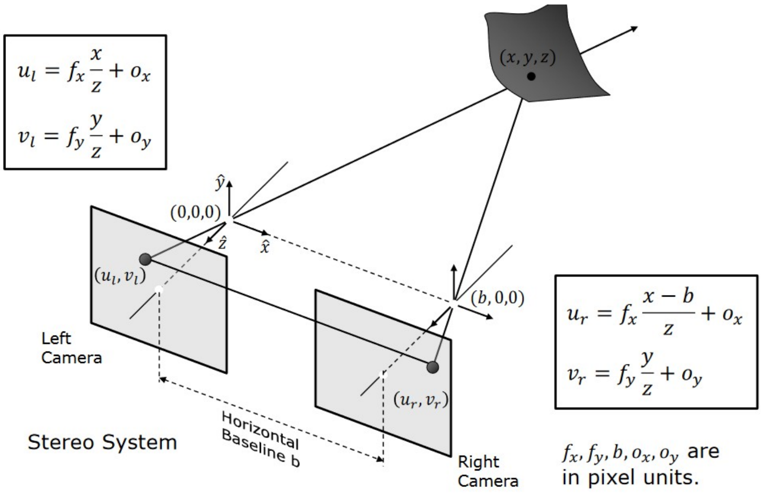

$u_l = f_x \frac{x}{z} + o_x$
$v_l = f_y \frac{y}{z} + o_y$
(x, y, z)
$\hat{y}$
(0,0,0)
$\hat{z}$
$\hat{x}$
(u_l, v_l)
Left Camera
$(b, 0, 0)$
$u_r = f_x \frac{x - b}{z} + o_x$
$v_r = f_y \frac{y}{z} + o_y$
(u_r, v_r)
Stereo System
Horizontal Baseline b
Right Camera
f_x, f_y, b, o_x, o_y are in pixel units.

FIGURE 2.6: Stereo camera setup for depth estimation. [Förstner and Wrobel, 2016]

By solving the following perspective projections:

$$(u_l, v_l) = \left(f_x \frac{x}{z} + o_x, f_y \frac{y}{x} + o_y\right) \qquad (u_r, v_r) = \left(f_x \frac{x-b}{z}, f_y \frac{y}{z} + o_y\right) \tag{2.11}$$

Where, b is a base offset between the two cameras, $v_l = v_r$

$$x = \frac{b(u_l - o_x)}{d} \qquad y = \frac{b f_x (v_l - o_y)}{f_y \cdot} \qquad z = \frac{b \cdot f_x}{d} \tag{2.12}$$

Where $d = u_l - u_r$ is called disparity.

Disparity is a difference between offsets of the projected points on both images.

Analyzing the equation for depth (z coordinate) we can conclude:

1. The closer the point to the camera the bigger the disparity is

2. To be able to reliably estimate the depth of a distant point, we need to have a large baseline

Taking into consideration the physical constraints of UAVs, it is impossible to have a large baseline for a stereo setup on UAVs. Despite it does not seem feasible to use a stereo camera setup for depth estimation on a UAV this method allows us to understand how it is possible to estimate depth from two images of the same scene taken by the same camera at different attitudes.

Another difficulty in using this method is finding of a corresponding point on the second image, especially in texture-less areas.

2.3 Interest points finding with Scale Invariant Feature Transform (SIFT) algorithm

To find disparity we need to find pairs of points on both images which belong to the same point in 3D space. To accomplish it we need a way to find some points on a picture which mark a distinct place in the image, these points are called Interest Points. As a single point can't provide enough distinct information, some area around the point (called a blob) is called an Interest Point. According to [Nayar, 2022a] blobs as interest points need to have:

- Exact location on image
- Determined size
- Determined orientation
- Have a unique enough signature which is independent of the size, orientation and brightness of the image

Once one finds interests points on both images and calculates their signature, they can run a matching algorithm to find close enough points on both pictures with a high probability that a couple of matched point belongs to one point in the 3D world.

Lowe [2004] proposed a Scale Invariant Feature Transform (SIFT) algorithm for finding interest points and creating point descriptors (signatures).

To find locations of the blobs at different scales, Lowe proposed to apply convolution of the input image with Normalized Laplacian of Gaussian

$$\sigma^2 \nabla^2 n_\sigma \tag{2.13}$$

with different values of σ. When a response of the convolution operation is above a threshold then the location and size of the blob are noted. To speed up calculations of convolution with Normalized Laplacian of Gaussian, Lowe proposed a trick where the image is processed with Gaussians at different scales then the difference of images processed at neighbouring scales is taken. This operation gives a good approximation of the desired more expensive operation. To remove the dependence on orientation the principal orientation of a blob is calculated and all blobs are rotated to have their orientation in the same direction. For this colour gradients of every pixel in the blob are calculated and normalized (to exclude the effect of lighting condition difference). Then Orientations are split into bins, the bin with the maximum number determines the principal orientation of the blob. The left part of Figure 2.7 shows the gradient calculation of pixels inside a blob, the right part depicts the histogram bins of the blob, and the largest bin is the principal orientation of the blob.

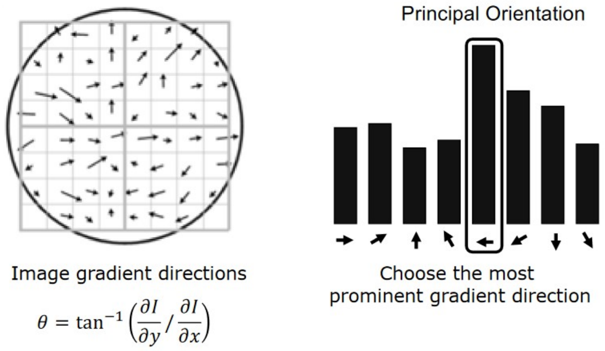

Principal Orientation
Image gradient directions
$\theta = \tan^{-1}\left(\frac{\partial I}{\partial y} / \frac{\partial I}{\partial x}\right)$
Choose the most prominent gradient direction

FIGURE 2.7: Finding orientation of a blob in SIFT detector. [Nayar, 2022b]

To create a signature of a blob, the blob is split into four quadrants, for every quadrant a histogram of orientations is created, and then histograms are concatenated to form a descriptor signature. This operation is presented in Figure 2.8.

The final descriptor is invariant to rotation, scale and brightness and allows comparison between descriptors with the use of Euclidean distance to determine how close two descriptors are to each other.

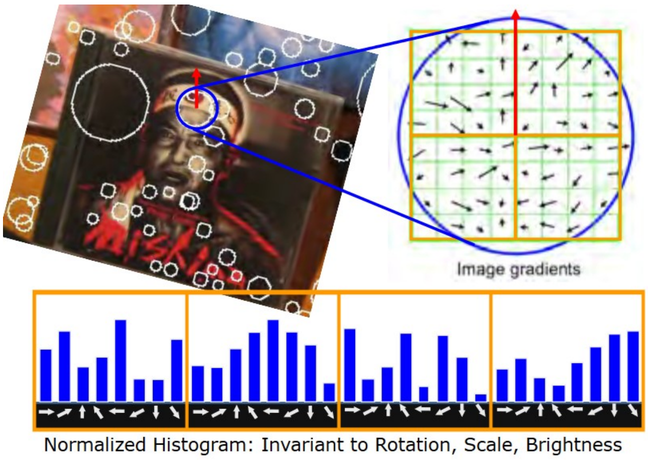


Image gradients
Normalized Histogram: Invariant to Rotation, Scale, Brightness

FIGURE 2.8: SIFT descriptor forming procedure. [Nayar, 2022b]

2.4 Depth estimation with Semi-global matching

The methods presented above allow to estimate depth at selected points only, resulting in a sparse depth map. To enable producing of dense maps, Hirschmueller [2008] proposed an algorithm for finding pixel-wise disparity which does not rely on window matching. This method showed good results on images with fine textures and depth discontinuities. The main idea is in minimising the total cost function for the whole disparity image D:

$$E(D) = \sum_{p}(C(p, D_p) + \sum_{q \in N_p} P_1 T[|D_p - D_q| = 1] + \sum_{q \in N_p} P_2 T[|D_p - D_q| > 1]) \qquad (2.14)$$

In this equation, the first term is a cost of pixel-wise disparities on the whole image, the second term adds a penalty for neighbouring pixels which have small disparity, and the third term adds a penalty for neighbouring pixels with big disparities.

Minimising this equation is an NP-hard problem, therefore the author proposed an optimized algorithm, which combines computations along 8 lines going through the picture and meeting at a pixel.

This method showed good results for depth estimation through disparity computation on aerial images but at a high computational cost.

2.5 Monocular Depth Estimation using Deep Learning

The most prominent feature of DL Depth Estimation (DE) is the ability to provide the estimation to all locations of the input image.

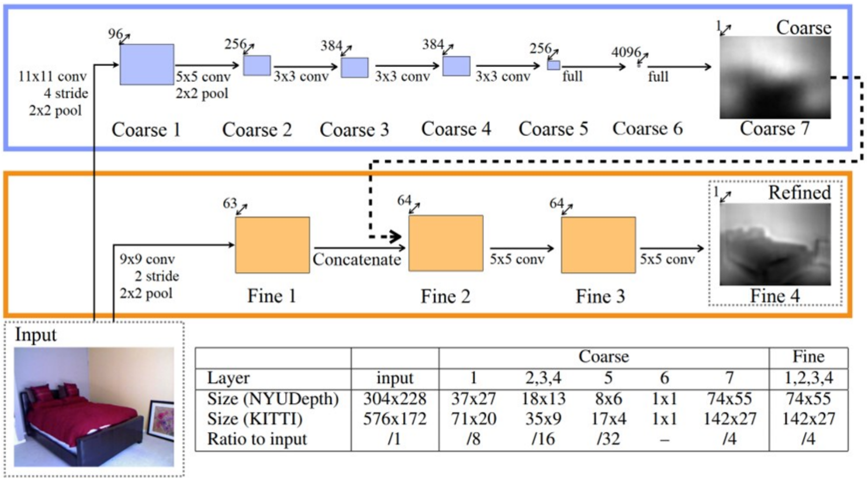
The community developed a wide range of DL models for DE to process single images, stereo images, sequences of images taken when the camera moves, and video. The last two data types can be used in single-image inference or sequence-to-sequence models.

The range of deep learning models developed for MDE can be split by the training approach: Supervised, Unsupervised and Semi-supervised.

This section is created with the support of work by Masoumian et al. [2022]

2.5.1 Supervised deep learning approach

In Supervised learning the dataset is made of tuples of the training data and Ground Truth (GT) depth for every pixel in the input image. The task of DE is formulated as regression. The training algorithm of such models is: predicting depth estimation of an input image and backpropagation of error signal calculated regarding GT. The first work in building a DL model for depth estimation is done by Eigen et al. [2014]. Their model is presented in Figure 2.9. It is composed of two networks: Coarse and Fine. The Coarse network produces a rough depth estimation and consists of 4 CNN layers and two fully connected layers. The Fine network is supposed to refine coarse estimation



		Coarse					Fine
Layer	input	1	2,3,4	5	6	7	1,2,3,4
Size (NYUDepth)	304x228	37x27	18x13	8x6	1x1	74x55	74x55
Size (KITTI)	576x172	71x20	35x9	17x4	1x1	142x27	142x27
Ratio to input	/1	/8	/16	/32	–	/4	/4

FIGURE 2.9: Model architecture by Eigen et al. [Eigen et al., 2014]

and is made of only CNN layers. While the Coarse network sees the full image, the Fine network looks only at one part of the image at a time with the addition of the full field of view from the Coarse network which is concatenated with output from layer 1 to form input for layer 2. Training of the network is done separately: coarse network first then fine network (the Coarse network does not propagate gradient during Fine network training). It is worth noticing, that the resulting depth map has a lower resolution than the input image due to the fully convolutional nature of the Fine network.

After Eigen's work, several different variants of this model were proposed to increase precision. One example is Li et al. [2015] work where the authors use Conditional random fields (CRF) for refining coarse DE to the pixel level. In their model, a coarse DE CNN network with two fully connected levels is connected to the CRF part to make the refinement of the super-pixel output of the Coarse network with the assistance of the corresponding super-pixel image patch. Despite enhancing precision, this work did not tackle the decrease in resolution of the final depth map.

Ummenhofer et al. [2016] recognized that a single image does not have enough information for precise depth estimation. Therefore, a single-image approach is bound to produce low precision on unseen data. Hence, the authors proposed a model depicted in Figure 2.10. The input data for this network is made of two images depicted from two positions (can be two successive frames recorded by a moving camera), the GT is DM for one image and rotation and transformation between these images. The main blocks of the networks are based on an encoder-decoder architecture similar to U-net [Frangi et al., 2015]. The output of the network is DM and ego-motion between the images. The model demonstrated better accuracy than the previous attempts. This paper does not provide detailed information about the layers, hence, it is impossible to estimate the computational cost of the proposed solution. Additionally, the authors did not evaluate the ranges of rotation and translation for which the architecture can provide reasonable results.

The recent work by Sheng et al. [2022] aimed to decrease the computational requirement of MDE with keeping the precision achieved in the previous works. The authors noticed, that for better depth estimation information from local neighbouring regions is not enough, hence, they proposed a Pyramid Scene Transformer block to enable capturing of interaction between multi-scale regions. The proposed architecture is presented in

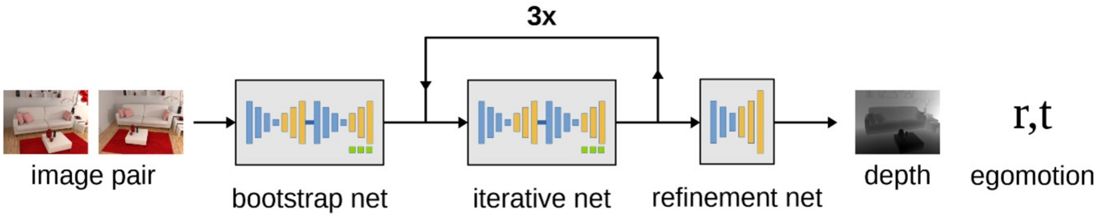


FIGURE 2.10: DeMoN model architecture for predicting depth map and ego-motion. [Ummenhofer et al., 2016]

Figure 2.11. The reported result demonstrated comparable performance with 1% FLOPs of prior models.

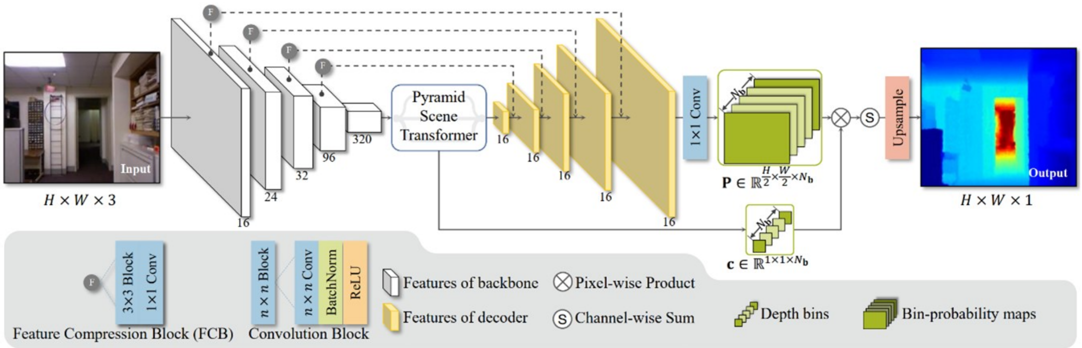


FIGURE 2.11: DANet network architecture. [Sheng et al., 2022]

Kumar et al. [2018] explored a way to enhance the precision of the depth estimation network with using of temporal information between consequent image frames. They used the idea of encoder-decoder and proposed Convolutional LSTM layers for the encoder part to use spacial information depicted in previous frames. The proposed model is shown in Figure 2.12.

The main disadvantage of supervised approaches is the requirement to provide ground truth (GT) depth information. Preparing such datasets is a laborious task which is difficult to achieve for high-altitude images.

2.5.2 Unsupervised Deep Learning approach

With the growth of the number of parameters, DL models require more data to overcome the overfitting effect. Therefore, the size of the dataset starts to play a crucial role in achieving high model performance. To enable the creation of larger datasets

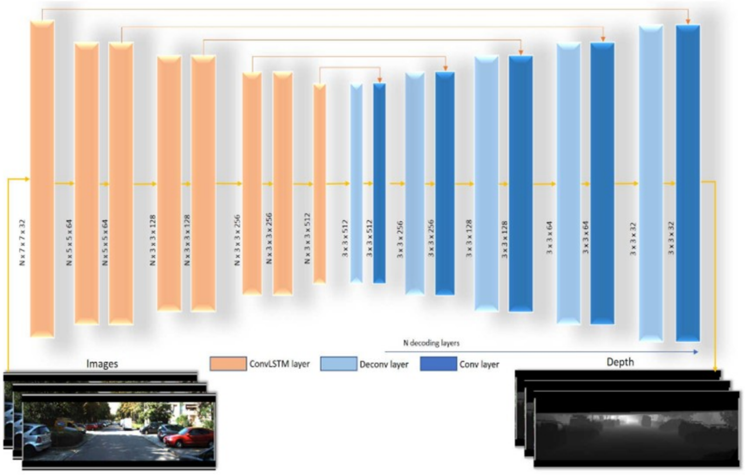


FIGURE 2.12: Convolutional LSTM encoder and decoder of DepthNet [Kumar et al., 2018]

and lift requirements for GT depth information, several research were made to develop unsupervised models for MDE.

The pioneering work was done by Garg et al. [2016] The authors used spacial information from a stereo image pair for calculating the error signal for backpropagation. The algorithm for calculating error is depicted in Figure 2.13. The model uses AlexNet

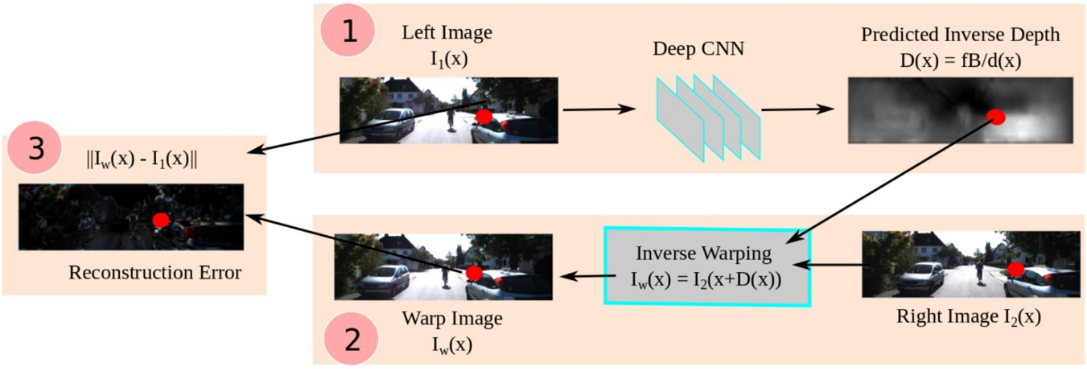


FIGURE 2.13: Auto-encoder setup for self-supervised learning. [Garg et al., 2016]

([Krizhevsky et al., 2017]) similar network for the encoder and fully-convolutional architecture for the decoder with a couple of skipping connections. During training left image is used for DE. As motion between the left and right images is known, this data is used for the reconstruction of the left image with the use of predicted DE and the right image. The authors acknowledged, that the method can be extended to be used on a sequence of images captured by a moving camera if translation between the images is known.

Work of Zhou et al. [2017] extended the approach used by Garg et al. [2016] by adding Pose Estimation Network to enable self-supervised learning on video. The proposed self-supervised pipeline is shown in Figure 2.14.

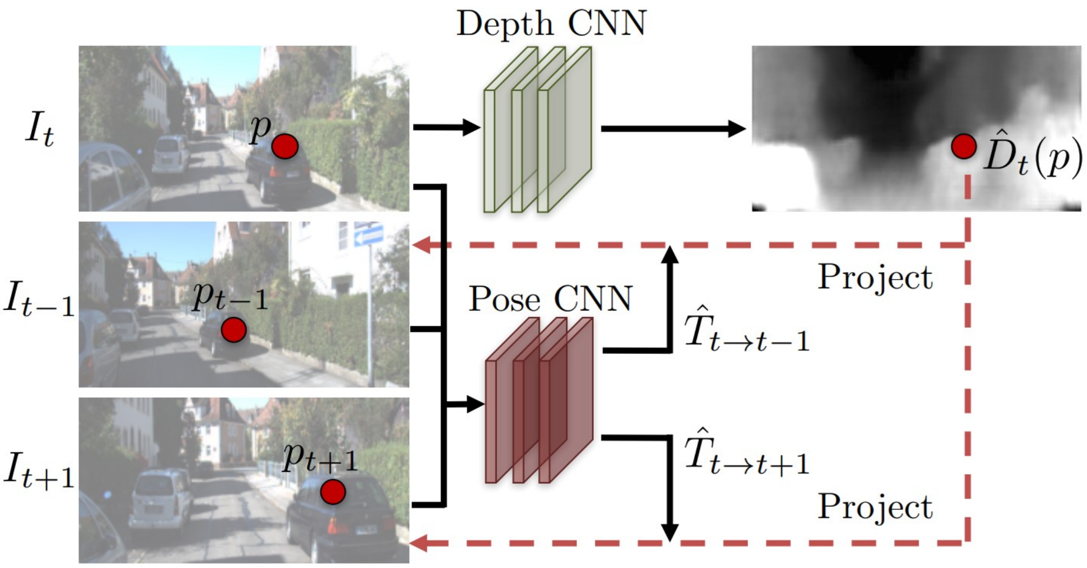


FIGURE 2.14: Pipeline for self-supervised learning of dept and pose. [Zhou et al., 2017]

In the proposed pipeline, the target frame I is used for depth estimation and neighbouring frames are used for pose estimation, the error is calculated by projecting the neighbouring frames with the use of depth information to reconstruct the target frame.

This idea was extended to be used for UAV recorded images by Hermann et al. [2020]. The proposed architecture allows to decrease in computational demand during inference time by using only the depth CNN part, while the Pose Estimation part is used only for training.

Masoumian et al. [2023] in their work followed the same approach proposed by Zhou et al. [2017] for capturing the error signal for supervised learning and focused on modification of the Depth Net decoder part. They used Graph Convolutional Network for depth map generation, see Figure 2.15.

Developing the idea proposed by Zhou et al. [2017], Makarov et al. [2022] exploited temporal information from a sequence of images (or video) by creating a sequence-to-sequence model for depth estimation. The authors extended the model proposed in [Zhou et al., 2017] by adding a sequence of Convolutional GRU blocks to the depth decoder. The extended model is presented in Figure 2.16.

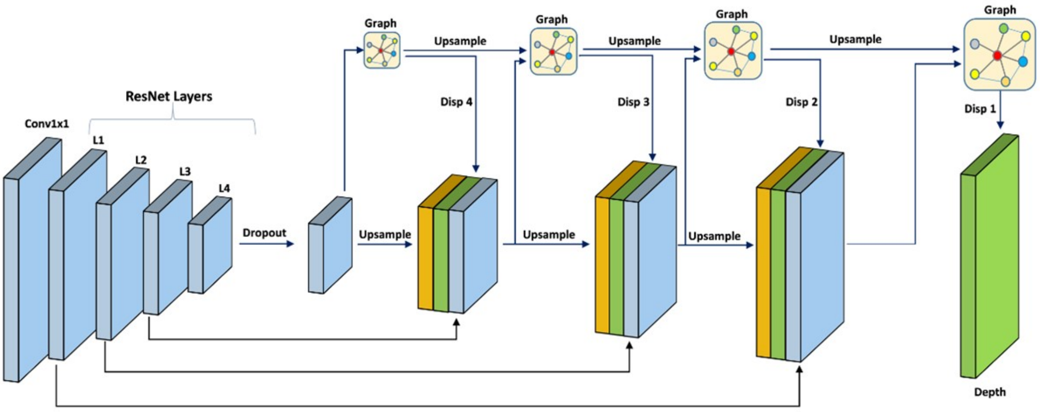


FIGURE 2.15: Depth net with GCN [Masoumian et al., 2023]

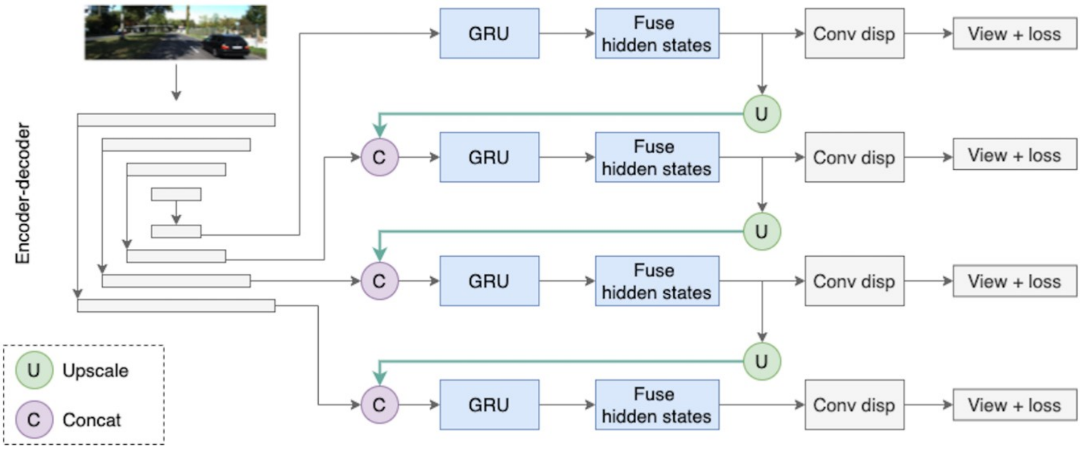


FIGURE 2.16: Sequence-to-sequence depth estimator with ConvGRU blocks. [Makarov et al., 2022]

2.5.3 Semi-supervised Deep Learning

To enable the training of DE models on cheap to acquire unlabeled data, researchers proposed several Semi-Supervised approaches, where other information is used for training ground truth depth maps for the whole input image. The methods can be split into two groups: using sparse data maps, training on synthetic data and transferring the model to real-world images for validation.

Sparse depth maps can be captured with the use of LIDAR. Kuznietsov et al. [2017] concentrated on using sparse depth maps during the learning stage only. Their model has similar to U-net architecture, which infers a depth map from a single image. The authors innovated in the development of loss calculation which uses images from binocular cameras and sparse depth maps, the components of the loss function are presented in Figure 2.17.

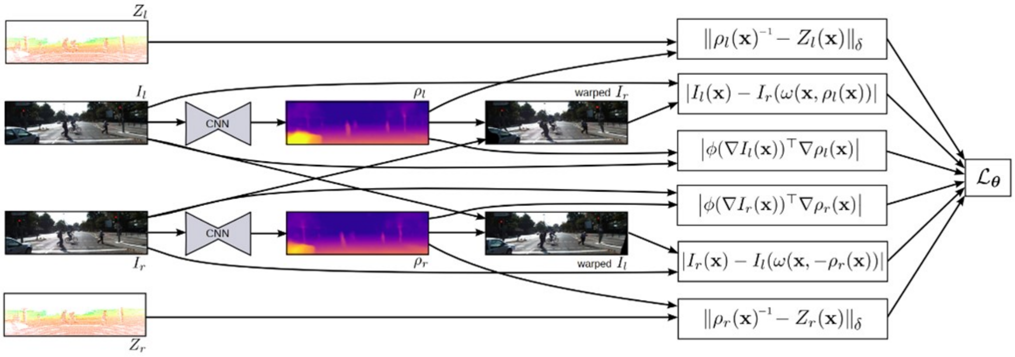

Z_l
I_l
CNN
ρ_l
warped I_r
I_r
CNN
ρ_r
warped I_l
Z_r
$\|\rho_l(\mathbf{x})^{-1} - Z_l(\mathbf{x})\|_\delta$
$|I_l(\mathbf{x}) - I_r(\omega(\mathbf{x}, \rho_l(\mathbf{x}))|$
$|\phi(\nabla I_l(\mathbf{x}))^\top \nabla \rho_l(\mathbf{x})|$
$|\phi(\nabla I_r(\mathbf{x}))^\top \nabla \rho_r(\mathbf{x})|$
$|I_r(\mathbf{x}) - I_l(\omega(\mathbf{x}, -\rho_r(\mathbf{x}))|$
$\|\rho_r(\mathbf{x})^{-1} - Z_r(\mathbf{x})\|_\delta$
$\mathcal{L}_\theta$

FIGURE 2.17: Semi-supervised loss function for MDE model by Kuznietsov et al. [Kuznietsov et al., 2017]

Another approach was proposed by Teixeira et al. [2020], the authors focused on the completion of sparse depth measurements from LIDAR with an image captured at the same time. The proposed architecture infers DM pixel-to-pixel from a captured image along with a confidence map. An additional requirement was the size of the inference network, to enable running it on board a UAV. Hence the inference network is relatively small, to facilitate training a bigger Loss network is developed, see Figure 2.18.

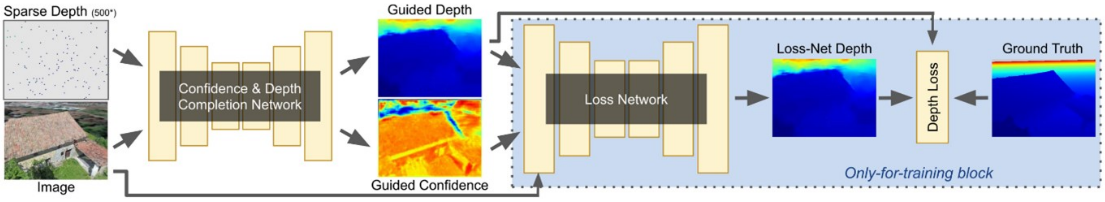


FIGURE 2.18: Training framework for DE from image and sparse LIDAR measurements. [Teixeira et al., 2020]

Another solution to remedy the lack of big datasets of GT DE is to use synthetic data for training and then transfer the model to do inference on real-world images. In this approach, to build a dataset, images are generated from 3D models along with full-depth information.

Kundu et al. [2018] proposed AdaDepth framework for transfer learning of DE from synthetic data to the real-world setup, see Figure 2.19. The architecture is trained in two steps, first, the blue channel is trained in fully supervised mode on synthetic data, and then the network is adjusted (purple channel) in an adversarial setup to real-world images without GT DM. During adjustment weights up to layer 5 are being changed as it was demonstrated in the paper, that these deep layers are more domain specific.

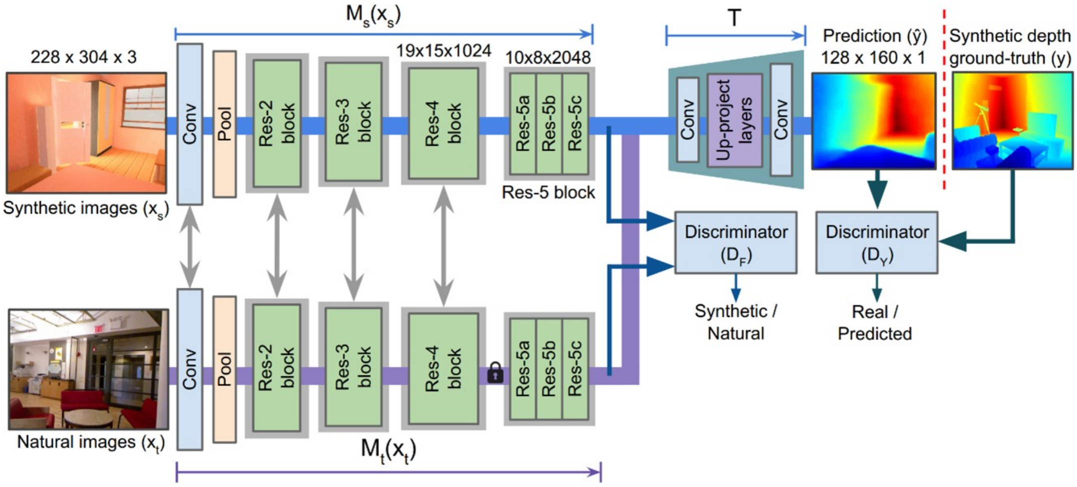


FIGURE 2.19: AdaDepth encoder-decoder architecture with adversarial setup for transfer learning. [Kundu et al., 2018]

2.6 Critical analysis

Depth maps represent 3D scene geometry and are an important input for UAV navigation and SLAM tasks. Both types of depth maps, sparse and dense, can be utilized by an onboard UAV navigation system as long as there are enough computational resources. Hence, we do not discard any approach from the start.

Most literature body concentrated on depth estimation is focused on indoor and ground view (e.g. autonomous car driving) use cases, therefore very few works pay attention to high-altitude depth estimation from a monocular camera.

Another peculiarity of UAV navigation is the availability of data from different onboard sensors (from IMU in the low-cost case) and control signals to motors issued by the navigation system captured simultaneously with images. Despite the noisy nature of this kind of data, numerous applications of Kalman filter proved that it is possible to reliably estimate the relative motion of a flying UAV. Fusing image data with the relative motion data might help to increase the quality of the depth estimation task. This is another area which is not explored in the literature.

During the flight, a UAV captures images continuously. The temporal information between in a sequence of images might be another source for quality enhancement of depth estimation. The literature body has several works covering sequence-to-sequence models for depth estimation, but we were not able to find any work that demonstrated how such models could perform on aerial images.

Chapter 3

Methodology

3.1 Overview

This project will investigate several approaches for depth estimation from a sequence of monocular images taken from a flying UAV. The work will be divided into several phases:

- Phase 1: Depth estimation from Interest points
- Phase 2: Pixel-wise depth estimation with Deep Learning models from a single image
- Phase 3: Pixel-wise depth estimation with a sequence-to-sequence Deep Learning model
- Phase 4: Using of Kalman filter for enhancing pixel-wise depth estimation from a Deep Learning model

The quality of depth estimation from different approaches will be compared and the feasibility of their use for UAV image-based on-board navigation system will be evaluated.

3.2 Project phase details

3.2.1 Phase 1: Depth estimation from Interest points

In this phase Interest points based method will be combined with Kalman filter to estimate depth of several points on images taken by a flying UAV.

For interest point detection and point matching between successive images, SIFT algorithm will be used. As the patent for SIFT algorithm expired in 2020 there is an implementation of the algorithm in OpenCV library free for research use.

When Interest Points in two successive images are detected and matched, one needs to estimate the distance the camera traveled between the corresponding two locations. To make this estimation more reliable a Kalman filter will be used.

After the travelled distance is estimated and points are matched, we can estimate depth of the matched Interest point pairs by calculating the disparity.

3.2.2 Phase 2: Pixel-wise estimation with Deep Learning model from a single image

During this phase, the DL approach will be evaluated for building dense depth map from a single image. As by using a single image without any prior knowledge of the scene, it is impossible to estimate real depth. Therefore the depth maps produced by the DL models will provide relative depth information.

Our preliminary test of existing DL models for depth estimation demonstrated the challenging nature of the goal of this phase. Hence, it is expected that a mix of supervised and unsupervised approaches will be used as both of them have benefits and drawbacks:

- Supervised DL: the use of a loss function that allows direct comparison of the estimated Depth Map with the Ground Truth should produce a better error signal for backpropagation. On the other hand, producing of ground truth is a challenging and expensive task for real-world images.
- Unsupervised DL: in this case, the loss function does not require ground truth. The error signal is calculated with the use of images from neighbouring positions.

As preliminary experiments showed, the existing models did not perform well on this task.

The first step in this phase is to train an existing architecture for DE on the data to get a baseline model for comparisons.

Possible activities for Phase 2 can be:

- Select an existing self-supervised model and experiment with modifications to its loss function
- Pre-train depth encoder in a self-supervised mode
- Pre-train depth estimator in supervised mode
- Introduce Pyramidal encoder in Depth Estimator part to exploit information from different scales

For DL training Tensorflow and PyTorch libraries will be used depending on the source code of the models used to start with.

3.2.3 Phase 3: Sequence-to-sequence DE DL model

This phase is optional and going to be executed if the time allocated for this project allows.

The main idea for this phase is to exploit temporal information between successive images taken by a flying camera. Potential architectural model blocks for a sequence-to-sequence model are: Convolutional LSTM (ConvLSTM) introduced in Shi et al. [2015] and Convolutional GRU (ConvGRU) introduced in Ballas et al. [2015]

Possible activities for this phase:

- Select a couple of models presented in the literature
- Train the models on aerial images
- Compare quality of depth estimation produced by sequence-to-sequence models and models from Phase 2

3.2.4 Phase 4: Application of Kalman filter to enhance DE of a DL model

This phase is also optional and will be done if the time frame for this project allows and the results of Phase 3 are promising.

A good candidate for a base DL model for this phase is a sequence-to-sequence model as this kind of models already exploits temporal information between successive images.

With Kalman Filter, it is expected to produce a good estimation of the distance the camera travelled between a couple of images. Therefore this information might be beneficial for:

- filtering erroneous changes in depth map while transitioning from one image to another
- potentially useful as input to depth estimation layers as an additional cue

3.3 Evaluation strategy

The main concern for this project is the ability to get GT depth values. The following strategies will be used for capturing of GT of input data:

1. Simulator-based data generation, e.g. with Gazebo simulator
2. Structure-from-Motion photogrammetric technique [1] to produce dense depth maps from images captured in the real world

From existing datasets, the one developed by Teixeira et al. [2020] looks promising for this work, but it has a limitation as the dataset provides sparse depth maps for images.

Additionally, we expect to get data from Technology Innovation Institute (TII) as they proposed the topic of the current work.

[1] https://en.wikipedia.org/wiki/Structure_from_motion, accessed: 26-03-2023

3.3.1 Metrics

To measure the performance of the final algorithm/model three-fold metrics will be used:

- The number of Float Point Operations (FLOPs) required for inference, to estimate the feasibility of running on-board a UAV.
- The amount of memory required for inference.
- The overall accuracy of DE.

To estimate how well the model made DE Eigen et al. [2014] formulated several metrics which were later commonly accepted. These metrics are:

- RMSE(linear)= $\sqrt{\frac{1}{|N|}\sum_{i\in N}\|d_i - d_i^*\|^2}$
- RMSE(log)= $\sqrt{\frac{1}{|N|}\sum_{i\in N}\|\log d_i - \log d_i^*\|^2}$
- Abs Relative difference= $\frac{1}{|N|}\sum_{i\in N}\frac{|d_i - d_i^*|}{d_i^2}$
- Squared Relative difference $= \frac{1}{|N|}\sum_{i\in N}\frac{\|d_i - d_i^*\|^2}{d_i^2}$
- Accuracies: % of d_i s.t. $\max\left(\frac{d_i}{d_i^*}\,\frac{d_i^*}{d_i}\right) = \delta < thr$
- RMSE(log, scale-invariant)= $\frac{1}{|N|}\sum_{i\in N}\left(\log d_i - \log d_i^* + \frac{1}{|N|}\sum_{j\in N}\left(\log d_j^* - \log d_j\right)\right)^2$

where d_i is the predicted depth value of i-th pixel, d_i^* is the ground truth depth value, N is the total number of pixels with depth value, *thr* is a threshold.

3.4 Requirements analysis

The aim of the project is to propose several algorithms for Depth Estimation from aerial images for use in the self-navigation of a drone.

The project is research-oriented. Its main objectives are:

1. Assess the state of the art of depth estimation from images
2. Develop an algorithm for depth estimation with the use of Interest Points with the use of Kalman filter for correction from a sequence of aerial images

3. Develop or extend already existing Deep Learning model for pixel-wise depth estimation from aerial images
4. Optionally develop or extend sequence-to-sequence Deep Learning model for pixel-wise depth estimation from a sequence of aerial images

3.4.1 Functional requirements

- The system should estimate depth from Interests Points with the use of Kalman filter to deal with noise and uncertainty.
- The system should produce a dense depth map (pixel-wise) from a single aerial image.
- The system could exploit temporal information from a sequence of aerial images for producing dense depth maps.
- The system could exploit state-space information of a UAV trajectory for enhancing depth maps.

3.4.2 Non-functional requirements

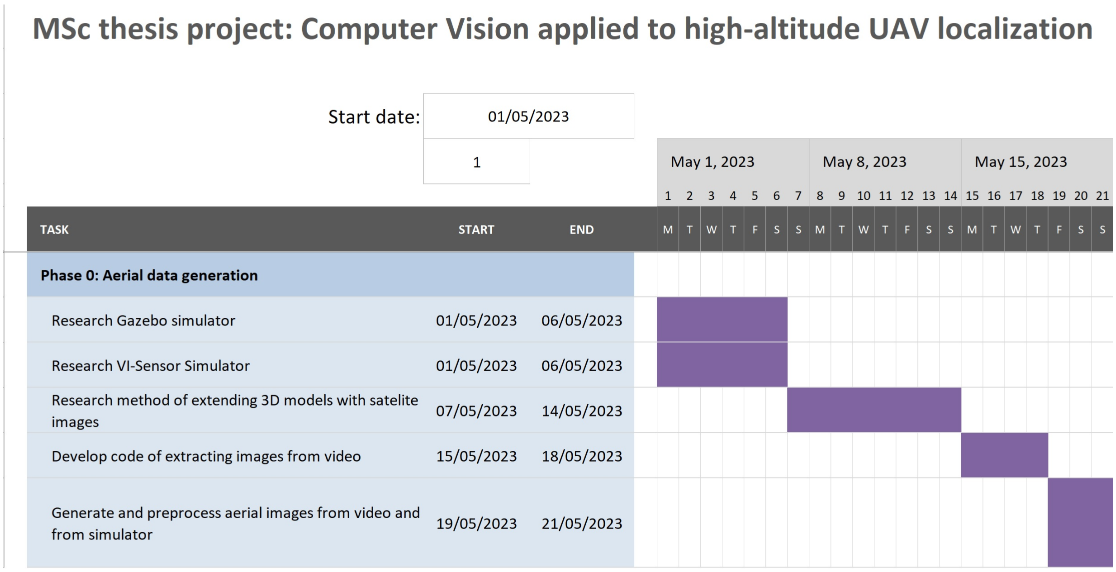
- The system would be able to run onboard a UAV for depth estimation for Interest points.
- The system would be able to produce a dense depth map on-board UAV CPU for navigation purposes.

Chapter 4

Project management

4.1 Work plan

The work schedule is presented below in the form of a Gantt chart. The chart is presented in four parts.



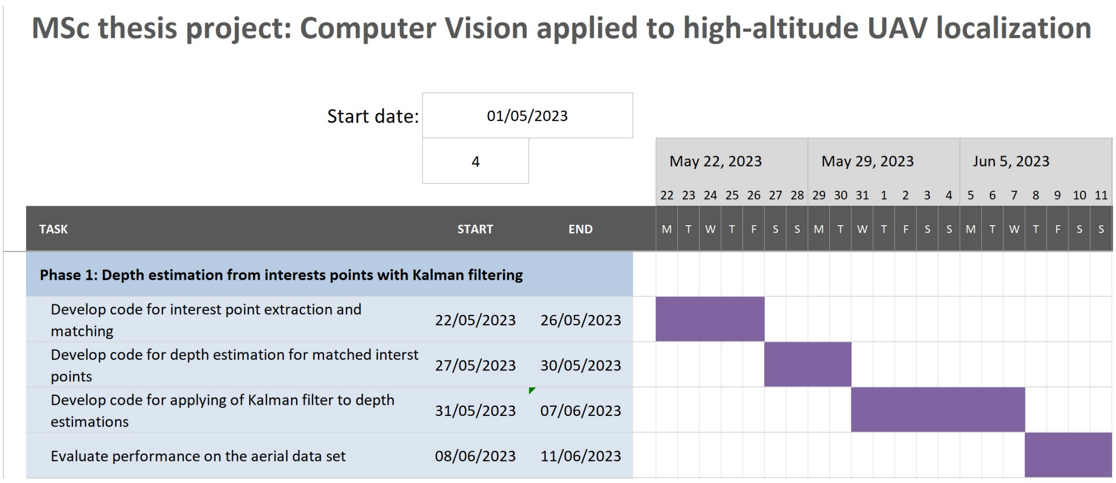
MSc thesis project: Computer Vision applied to high-altitude UAV localization

Start date: 01/05/2023

1

TASK	START	END
Phase 0: Aerial data generation		
Research Gazebo simulator	01/05/2023	06/05/2023
Research VI-Sensor Simulator	01/05/2023	06/05/2023
Research method of extending 3D models with satelite images	07/05/2023	14/05/2023
Develop code of extracting images from video	15/05/2023	18/05/2023
Generate and preprocess aerial images from video and from simulator	19/05/2023	21/05/2023

FIGURE 4.1: Project schedule. Part 1.



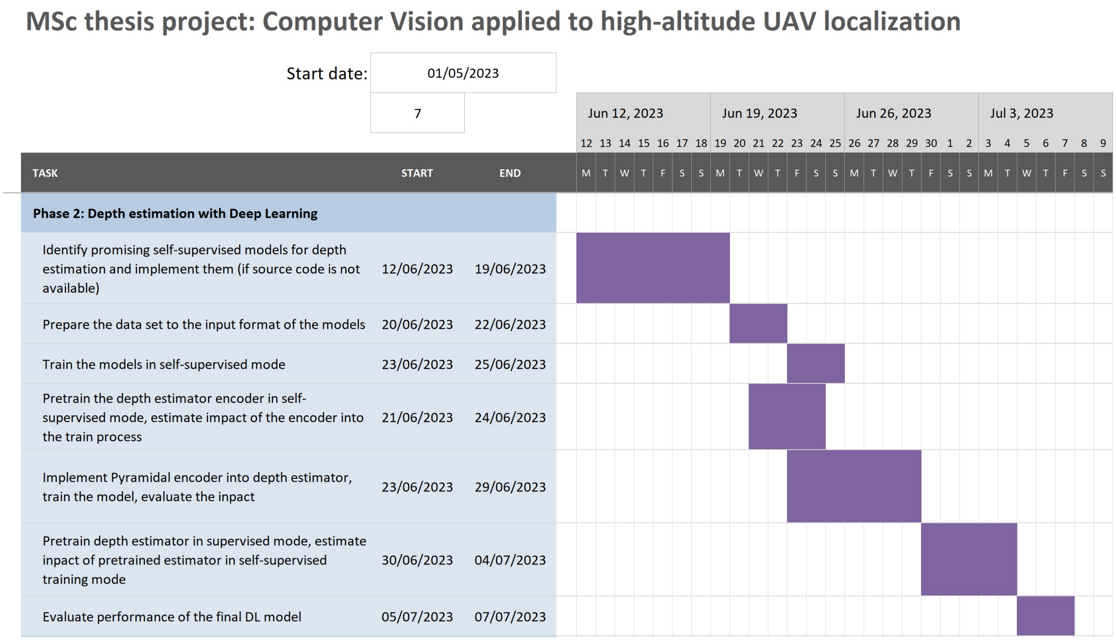
MSc thesis project: Computer Vision applied to high-altitude UAV localization

Start date: 01/05/2023

4

TASK	START	END
Phase 1: Depth estimation from interests points with Kalman filtering		
Develop code for interest point extraction and matching	22/05/2023	26/05/2023
Develop code for depth estimation for matched interst points	27/05/2023	30/05/2023
Develop code for applying of Kalman filter to depth estimations	31/05/2023	07/06/2023
Evaluate performance on the aerial data set	08/06/2023	11/06/2023

FIGURE 4.2: Project schedule. Part 2.



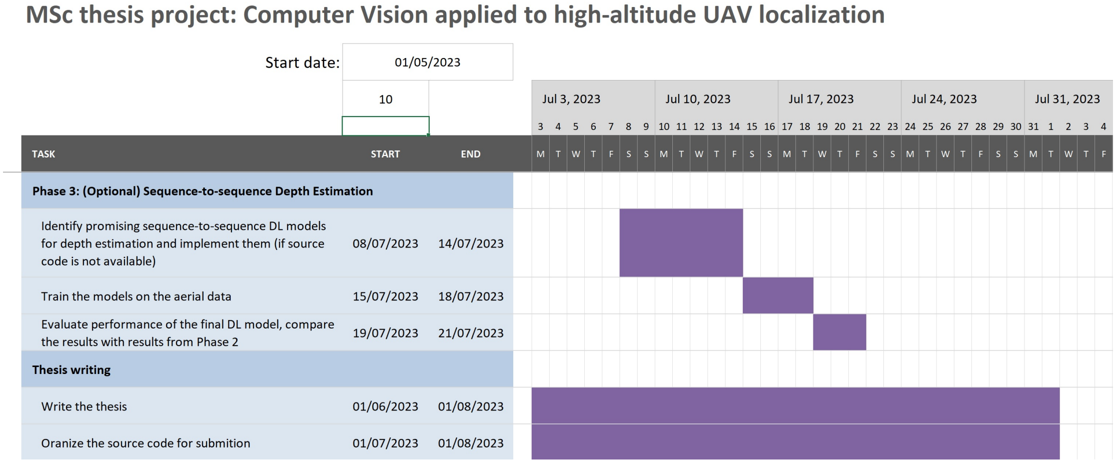
MSc thesis project: Computer Vision applied to high-altitude UAV localization

Start date: 01/05/2023

7

TASK	START	END
Phase 2: Depth estimation with Deep Learning		
Identify promising self-supervised models for depth estimation and implement them (if source code is not available)	12/06/2023	19/06/2023
Prepare the data set to the input format of the models	20/06/2023	22/06/2023
Train the models in self-supervised mode	23/06/2023	25/06/2023
Pretrain the depth estimator encoder in self-supervised mode, estimate impact of the encoder into the train process	21/06/2023	24/06/2023
Implement Pyramidal encoder into depth estimator, train the model, evaluate the inpact	23/06/2023	29/06/2023
Pretrain depth estimator in supervised mode, estimate inpact of pretrained estimator in self-supervised training mode	30/06/2023	04/07/2023
Evaluate performance of the final DL model	05/07/2023	07/07/2023

FIGURE 4.3: Project schedule. Part 3.



MSc thesis project: Computer Vision applied to high-altitude UAV localization

Start date: 01/05/2023

10

TASK	START	END
Phase 3: (Optional) Sequence-to-sequence Depth Estimation		
Identify promising sequence-to-sequence DL models for depth estimation and implement them (if source code is not available)	08/07/2023	14/07/2023
Train the models on the aerial data	15/07/2023	18/07/2023
Evaluate performance of the final DL model, compare the results with results from Phase 2	19/07/2023	21/07/2023
Thesis writing		
Write the thesis	01/06/2023	01/08/2023
Oranize the source code for submition	01/07/2023	01/08/2023

FIGURE 4.4: Project schedule. Part 4.

4.2 Risks analysis

This section presents an analysis of the risks which can affect the progress of the project. The possible risks and mitigation strategies are presented in Table 4.1.

Risk Description	**Likelihood**	**Impact**	**Mitigation Strategy**
Data or Source code loss	L	H	Keep data and source code backups in HWU cloud storage
DL model takes too much time or memory to train	H	H	Decrease image resolution, decrease model complexity, look for a GPU with bigger RAM
Requirements change	M	H	Organize the project development in iterations
Project plan or scope change	M	H	Change the plan in coordination with the supervisor
Supervisor or Student is ill	L	H	Switch from face-to-face communication to Teams calls and emails

TABLE 4.1: Risk analysis

4.3 Professional, Legal, Ethical and Social Issues

4.3.1 Professional Issues

The work done under the current project will adhere to the British Computer Society Code of Conduct rules. The source code will be created with the use of modern software development practices.

4.3.2 Legal Issues

During the work on the project, several existing libraries and program products will be used. The use of code and software will be under strict adherence to the licensing restrictions of the used components. As the work on this project is a joint effort of Heriot-Watt University and Technology Innovation Institute, there is a Non-disclosing agreement signed. The conditions of the agreement will be respected and no confidential information will be published without written permission of the source of the information.

4.3.3 Ethical and Social Issues

The project and the results of the project do not rise any ethical or social issues itself. But the project aims to contribute to the advance of the technology of self-navigating drones.

If to consider the technology of self-navigating drones itself, several ethical and social issues can be formulated. Here is a non-complete list of possible concerns ([Kurlekar, 2019], [Sandbrook, 2015]):

1. Drones can provide high-quality images of areas that are inaccessible from the ground view. This rises privacy concerns.
2. Widespread use of drones for delivery can cause job displacement.
3. The technology itself can be easily used for legal and illegal activities.
4. Drones can pose safety risks to people and animals.

At the same time, this technology can bring high-value benefits to society such as (the list is non-complete):

1. Enhancements in delivery,
2. Help in disaster recovery,
3. Law-enforcement activities,
4. Decrease in carbon footprint.

Hence, widespread deployment of the technology must be preceded by thorough thought-through and development of safeguards in law with enforcing of capability limits in the technology itself.

Bibliography

Ballas, N., Yao, L., Pal, C. J., and Courville, A. C. (2015). Delving deeper into convolutional networks for learning video representations. *CoRR*, abs/1511.06432.

Cherian, A., Andersh, J., Morellas, V., Papanikolopoulos, N., and Mettler, B. (2009). Autonomous altitude estimation of a uav using a single onboard camera. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3900–3905.

Conte, G. and Doherty, P. (2008). An integrated uav navigation system based on aerial image matching. In *2008 IEEE Aerospace Conference*, pages 1–10.

Eigen, D., Puhrsch, C., and Fergus, R. (2014). Depth map prediction from a single image using a multi-scale deep network. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, page 2366–2374, Cambridge, MA, USA. MIT Press.

Forsyth, D., Ponce, J., Mukherjee, S., and Bhattacharjee, A. (2012). *Computer vision: 2nd ed.* Pearson, Boston, MA.

Frangi, A. F., Hornegger, J., Navab, N., and Wells, W. M. (2015). U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, volume 9351 of *Lecture Notes in Computer Science*, pages 234–241. Springer International Publishing AG, Switzerland.

Förstner, W. and Wrobel, B. (2016). *Photogrammetric Computer Vision.* Springer International Publishing AG., Cham, Switzerland.

Garg, R., Kumar, B. V., Carneiro, G., and Reid, I. D. (2016). Unsupervised cnn for single view depth estimation: Geometry to the rescue. In *European Conference on Computer Vision*.

Grewal, M. S. and Andrews, A. P. (2015). *Kalman filtering theory and practice using MATLAB, fourth edition.* Wiley, Hoboken, NJ, 4th ed. edition.

Gupta, A. and Fernando, X. (2022). Simultaneous localization and mapping (slam) and data fusion in unmanned aerial vehicles: Recent advances and challenges. *Drones*, 6(4).

Gyagenda, N., Hatilima, J., Roth, H., and Zhmud, V. (2022). A review of gnss-independent uav navigation techniques. *Robotics and Autonomous Systems*, 152:104069.

Hermann, M., Ruf, B., Weinmann, M., and Hinz, S. (2020). Self-supervised learning for monocular depth estimation from aerial imagery. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, V-2-2020:357–364.

Hirschmueller, H. (2008). Stereo processing by semiglobal matching and mutual information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2(30):328–341.

Krizhevsky, A., Sutskever, I., and Hinton, G. (2017). Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90.

Kumar, A. C., Bhandarkar, S. M., and Prasad, M. (2018). Depthnet: A recurrent neural network architecture for monocular depth prediction. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 396–3968.

Kundu, J. N., Uppala, P. K., Pahuja, A., and Babu, R. V. (2018). Adadepth: Unsupervised content congruent adaptation for depth estimation. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2656–2665.

Kurlekar, S. (2019). Autonomous drones come with challenges and great potential. *TechTarget web site.* Accessed: 2023-03-19.

Kuznietsov, Y., Stuckler, J., and Leibe, B. (2017). Semi-supervised deep learning for monocular depth map prediction. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2215–2223, Los Alamitos, CA, USA. IEEE Computer Society.

Li, B., Shen, C., Dai, Y., van den Hengel, A., and He, M. (2015). Depth and surface normal estimation from monocular images using regression on deep features and hierarchical crfs. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1119–1127.

Liu, Y., Tao, J., Kong, D., Zhang, Y., and Li, P. (2022). A visual compass based on point and line features for uav high-altitude orientation estimation. *Remote Sensing*, 14(6).

Lowe, D. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60:91–.

Makarov, I., Bakhanova, M., Nikolenko, S., and Gerasimova, O. (2022). Self-supervised recurrent depth estimation with attention mechanisms. *PeerJ Computer Science*, 8:e865.

Masoumian, A., Rashwan, H. A., Abdulwahab, S., Cristiano, J., Asif, M. S., and Puig, D. (2023). Gcndepth: Self-supervised monocular depth estimation based on graph convolutional network. *Neurocomputing*, 517:81–92.

Masoumian, A., Rashwan, H. A., Cristiano, J., Asif, M. S., and Puig, D. (2022). Monocular depth estimation using deep learning: A review. *Sensors*, 22(14).

Nayar, S. K. (2022a). *Image Formation. First Principles of Computer Vision.* Columbia University, New York.

Nayar, S. K. (2022b). *SIFT Detector. First Principles of Computer Vision.* Columbia University, New York.

Sandbrook, C. (2015). The social implications of using drones for biodiversity conservation. *Ambio*, 44(Suppl 4):S636–S647.

Sheng, F., Xue, F., Chang, Y., Liang, W., and Ming, A. (2022). Monocular depth distribution alignment with low computation. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 6548–6555.

Shi, X., Chen, Z., Wang, H., Yeung, D.-Y., Wong, W.-k., and Woo, W.-c. (2015). Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, page 802–810, Cambridge, MA, USA. MIT Press.

Teixeira, L., Alzugaray, I., and Chli, M. (2018). Autonomous aerial inspection using visual-inertial robust localization and mapping. In Hutter, M. and Siegwart, R., editors, *Field and Service Robotics*, pages 191–204, Cham. Springer International Publishing.

Teixeira, L., Oswald, M. R., Pollefeys, M., and Chli, M. (2020). Aerial single-view depth completion with image-guided uncertainty estimation. *IEEE Robotics and Automation Letters*, 5:1055–1062.

Ummenhofer, B., Zhou, H., Uhrig, J., Mayer, N., Ilg, E., Dosovitskiy, A., and Brox, T. (2016). Demon: Depth and motion network for learning monocular stereo. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5622–5631.

Zhou, T., Brown, M., Snavely, N., and Lowe, D. G. (2017). Unsupervised learning of depth and ego-motion from video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.