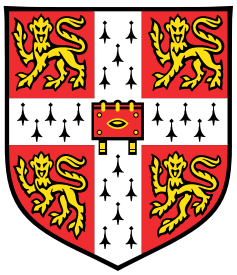
Approximate Depth Estimation in Colonoscopy Images



Gwangbin Bae

Department of Engineering
University of Cambridge

This dissertation is submitted for the degree of
Master of Philosophy

Hughes Hall May 2020

Abstract

This thesis introduces a novel method of estimating depth from monocular colonoscopy images. Our work is motivated by the need for cheap and efficient method of estimating the 3D geometry of a human colon. Such method can be used to identify the size and shape of the polyps and determine the safest path along which to move the endoscope. This would improve the accuracy and safety of the colonoscopy procedure and enable earlier detection of the colorectal cancer.

Current methods in colonoscopy depth estimation utilise stereo cameras [1], photometric stereo [30], depth sensors [5], and structured light [34]. In order to apply these methods, specially designed light source or sensors should be added to the colonoscopy device. Such modifications increase cost and size of the device, thereby making colonoscopy more expensive and dangerous. Our method, on the other hand, can estimate depth from a single monocular image. This method does not require additional sensors and therefore can be applied to the existing colonoscopy procedure.

Our monocular depth estimation framework consists of three steps. Firstly, sparse relative depth is generated for short sequences of colonoscopy images, using structure from motion [36]. Then, the obtained ground truth is used to train a pixel-wise depth estimation network. Since the ground truth depth is in arbitrary scale, scale-invariant training loss is introduced to effectively train the model. Lastly, the predictions made on unseen images are scale-matched to the corresponding ground truth to allow both quantitative and qualitative evaluation. The average relative error (REL) and root mean square error (RMSE) on the test set are 0.0566 and 9.99, respectively. The absolute scale RMSE, estimated for a subset of test sequences, is $2.7mm$.

Our contributions include (1) an accurate pixel-wise depth prediction model trained with the scale-invariant loss, (2) large-scale dataset of ground truth relative depth for monocular colonoscopy images, and (3) quantitative and qualitative evaluation of the proposed method.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

1.1 Objective

The objective of this project is to build a monocular depth estimation framework for colonoscopy images. The framework should not require modification of the device, and should be able to make dense pixel-wise depth prediction for a single monocular colonoscopy image.

1.2 Motivation

Colorectal cancer is the second most common cause of cancer deaths [16]. In 2018, 1.8 million new cases and 0.9 million deaths were reported [16]. Despite such high risk, the five year survival rate is over 97% if the cancer is detected at its earliest stage [3]. It is therefore advised to do colonoscopy on a regular basis. During colonoscopy procedure, the physician identifies and removes polyps to examine whether they are benign or pre-cancerous (see Figure 1.1 for schematic explanation of the colonoscopy procedure).

Building a monocular depth estimation framework can help address the following challenges in the human-operated colonoscopy.

- **Polyp size and shape estimation.** The risk of a polyp can be assessed by its size and shape [7]. However, it is difficult for a human expert to understand the 3D geometry of the colon surface from 2D images. Monocular depth estimation framework can be used to recover the 3D reconstruction of the local colon surface, helping the physician to identify the polyps with high risk.

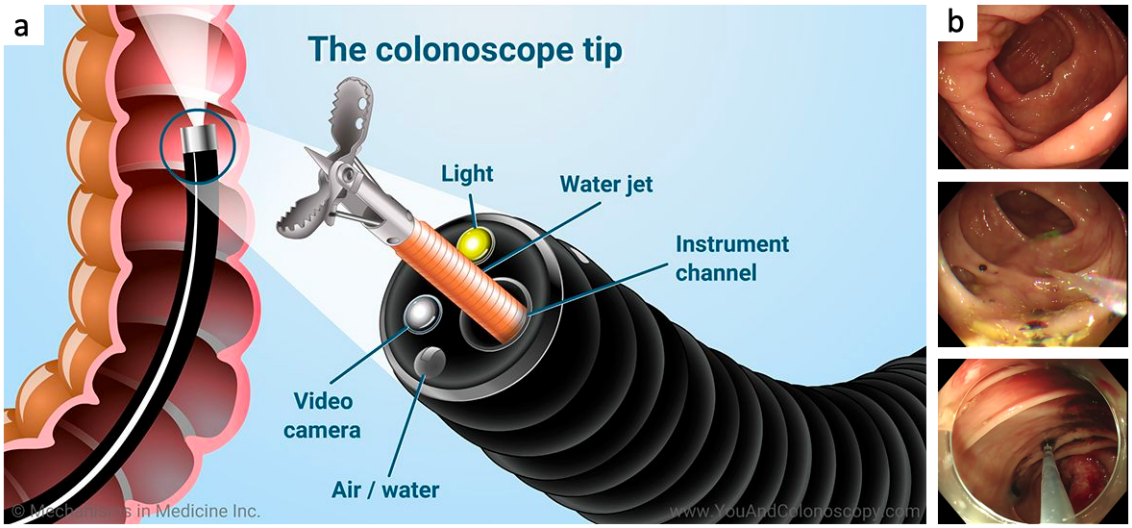


Fig. 1.1 Figure (a) shows a schematic diagram of a modern colonoscope [38]. Typically, the device is 11 to 13*mm* in diameter, and 1.3 to 1.5*m* in length. The tip of the device contains a camera, a light source, a nozzle for water jet, a channel for air and water, and an instrument channel. The water jet is used to cleanse the colon surface. The air channel inflates the colon for better visibility. If needed, the air and liquid are sucked through the same channel. Lastly, the instrumental channel is used to remove and retrieve the identified polyps. Three images in (b) show typical views from the camera during colonoscopy.

- **Coverage estimation.** It is difficult for a human expert to check if every part of the colon surface is scanned. Un-scanned regions can be identified by connecting the local 3D reconstructions estimated by the framework.

- **Assisted control.** It is difficult for a human expert to manually control the device while looking at a screen (i.e. under the absence of depth). Inaccurate control during polyp removal can result in colon perforation which itself can lead to death [21]. Monocular depth estimation framework can be used to assist the physician's control of the device, for example, by preventing the colonoscope tip from hitting the colon wall.

- **Semi-automated colonoscopy.** Since it requires considerable amount of training for a physician to be able to perform colonoscopy, the number of certified physicians is in short supply, making it difficult for people with no insurance coverage, or those in developing countries to receive colonoscopy. Monocular depth estimation framework, with the help of appropriate decision making algorithms, can semi-automate the device navigation and scanning, making the procedure easier for the physicians, and more accessible for the patients.

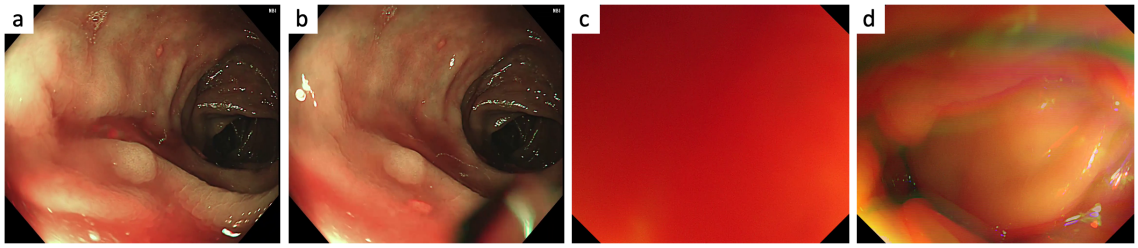


Fig. 1.2 Figure (a) and (b) show an example of a non-rigid colon motion. Note that the time difference between the two images is one second. Figure (c) and (d) are examples of colonoscopy images that have small number of unique visual features. In (c), the camera is very close to the colon wall. In (d), the camera is moving quickly, creating huge motion blur.

1.3 Challenges

Following are the challenges in building a depth estimation framework for colonoscopy images. Examples of these challenges are provided in Figure 1.2.

- **Limited sensors.** Due to cost and strict clinical regulation, it is difficult to add additional camera or sensors to the colonoscopy device. Under such limitation, the depth estimation framework should only use monocular cues.
- **Non-rigid deformation.** Colon surface continuously moves in a non-rigid manner, and is often deformed by the colonoscope. Since the geometry of the surface is not consistent between images, it is difficult, if not impossible, to estimate a single 3D reconstruction that can explain all the images.
- **Feature-less surface.** Colon surface is smooth and contains small number of unique features. It is therefore difficult to identify many feature correspondences between images. Thus, it is challenging to obtain dense reconstruction of the surface.

1.4 Approach

Due to non-rigid colon motion and lack of unique visual features, structure from motion [36] has been considered inappropriate for colonoscopy images. However, in this work we show that, while it is difficult to reconstruct the entire colon, it is possible to obtain sparse reconstruction of the local colon surface based on a short sequence of images and use this to train a monocular depth estimation network. Our approach, illustrated in Figure 1.3, consists of three steps.

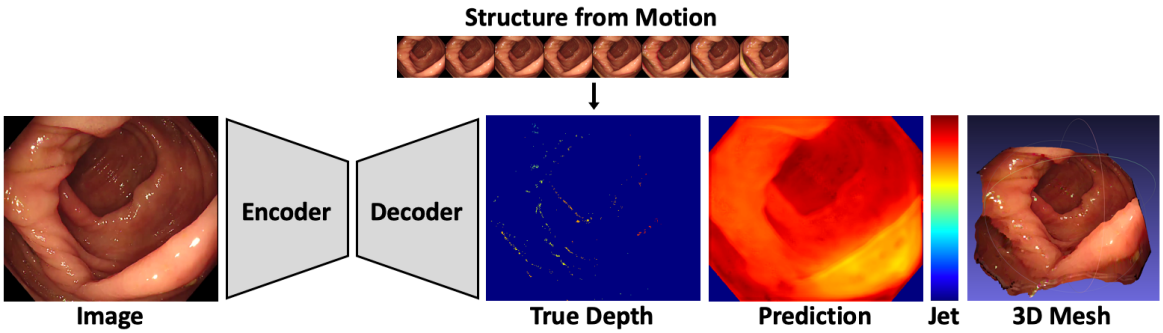


Fig. 1.3 This figure shows a simple schematic of our approach. For the given colonoscopy images, sparse relative depth is obtained using structure from motion. Despite the sparsity of the ground truth, the model is able to make dense prediction after training. Then, the model output can be used to create a 3D reconstruction of the local colon surface. Both the true depth and prediction are plotted using the Jet colour scheme.

- Firstly, the colonoscopy videos are divided into sequences of 8 frames. The time-difference between the consecutive frames in a sequence is selected to be small enough to reduce the effect of the non-rigid colon motion, but long enough to ensure sufficient change in the camera view. Then, structure from motion [36] is applied to each sequence to estimate its 3D reconstruction. To achieve more accurate results, we force the algorithm to only consider the features that appear in all 8 frames. We also discard any sequence for which the algorithm failed to estimate all 8 camera poses. The obtained sparse reconstruction is then projected to the images (using the estimated camera poses) to yield the sparse pixel-wise true depth, in arbitrary scale.

- Secondly, the generated ground truth is used to train a convolutional encoder-decoder network with single output channel. To account for the arbitrary scale and sparsity of the data, we introduce a sparse scale-invariant loss function. This function adjusts the scale of the prediction to match that of the true depth, and then computes the squared distance at the pixels with ground truth.

- Lastly, once the model is trained, it is tested on unseen images. The quality of the prediction is assessed after matching the scale of the prediction to that of the corresponding ground truth.

We evaluated our method on internal dataset generated from raw colonoscopy videos. The root mean square error (RMSE) and average relative error (REL) measured on the test set, are 9.99 and 0.0566, respectively. The absolute scale RMSE, estimated for a subset of test sequences, is 2.7mm. This suggests that the model prediction, if properly scaled, can be used to measure the polyp size and assist the device navigation.

1.5 Contributions

- **Depth estimation model.** We present a pixel-wise depth estimation model that can make dense prediction on depth given a single monocular image. We introduce the scale-invariant training loss and show that it results in better accuracy, compared to other training losses.
- **Ground truth dataset.** We introduce a dataset of $67,840$ monocular colonoscopy images and the corresponding ground truth relative depth. We show that structure from motion, with appropriate filtering and constraints, can estimate accurate reconstruction of the local colon surface.
- **Evaluation protocol.** We introduce both quantitative and qualitative protocols for evaluating the performance of the depth estimation model.

1.6 Outline

The remaining chapters are organized as follows. Chapter 2 introduces the works relevant to this project. Chapter 3 provides a detailed explanation on how the true depth can be obtained and used to train a pixel-wise depth estimation network. Chapter 4 explains the details of the experiment setup. Chapter 5 introduces the key experiments and their results. Lastly, Chapter 6 summarises the contributions of the project and discusses the possible extensions.

Chapter 2

Related Work

This chapter introduces the related work. The first section provides an overview of other approaches in colonoscopy depth estimation. The second section explains the structure from motion pipeline in detail. The last section explains how pixel-wise depth estimation can be performed by neural networks.

2.1 Depth Estimation in Colonoscopy Images

This section introduces the previous attempts in colonoscopy depth estimation. These methods can be categorized into two groups - measurement-based methods and learning-based methods.

2.1.1 Measurement-based Methods

Measurement-based methods utilise additional information, such as the sensor readings or prior knowledge, to estimate the depth.

Schmalz et al. [34] used structured light to measure the true depth. Ring shaped patterns of different colours are projected onto the surface and the reflected pattern is decoded to infer the 3D geometry of the surface. Hou et al. [15] improved this approach by developing active stereo-vision system, where the device switches between two viewpoints to find the view that gives the best result.

Chen et al. [5] attached distance sensors at the end of the endoscope to measure the distance of the device from the colon wall and identify the safe navigation path (i.e. along the central axis of the colon). In this setup, three distance sensors are facing distinct directions perpendicular to the moving direction. Each sensor emits light and measures the amount of the reflection to infer the depth in that direction.

Parot et al. [30] introduced photometric stereo endoscopy. Photometric stereo estimates the geometry of the surface from more than two images taken at the same camera pose but under different lighting conditions. To achieve this, an additional light source was attached to the colonoscope.

All of the above methods require the modification of the colonoscopy device, and are therefore costly. Furthermore, due to clinical regulation, these methods could not be tested on human colon. The evaluation of the methods was thus limited to virtual colon, colon phantom, or porcine colon, all of which are not representative of the human colon.

Hong et al. [14] tried to measure depth from the brightness of the pixels. This method does not require the modification of the device. However, the equation correlating the brightness of the pixel and its depth depends on several restrictive assumptions (e.g. the colon surface is Lambertian). Quantitative evaluation of the method was only made on synthetic colon, which had smooth surface with no blood vessels, unlike the real colon.

2.1.2 Learning-based Methods

Alternatively, the depth can be predicted using a model trained on readily-available dataset. Both supervised and unsupervised learning-based methods were attempted to achieve colonoscopy depth estimation.

Mahmood and Durr [23] proposed a neural network architecture based on deep convolutional neural network (CNN) and conditional random field (CRF). This pixel-wise depth estimation network was trained on 200,000 images generated from synthetic colon and CT colonoscopy. Due to the lack of ground truth, the model could not be trained or tested on human data.

Zhou et al. [39] introduced an unsupervised learning framework for monocular depth and ego-motion estimation. When given a sequence of monocular images, the model estimates the depth and camera pose that minimises the photometric loss. Itoh et al. [17] applied this approach to colonoscopy images to perform binary classification of the polyp sizes (larger than $10mm$ or not). This method does not require ground truth depth and can be trained only with monocular colonoscopy videos. However, since the colon surface is smooth and has small variation in colour, the photometric loss can be small for incorrect depth and camera pose, making it challenging for the model to learn depth.

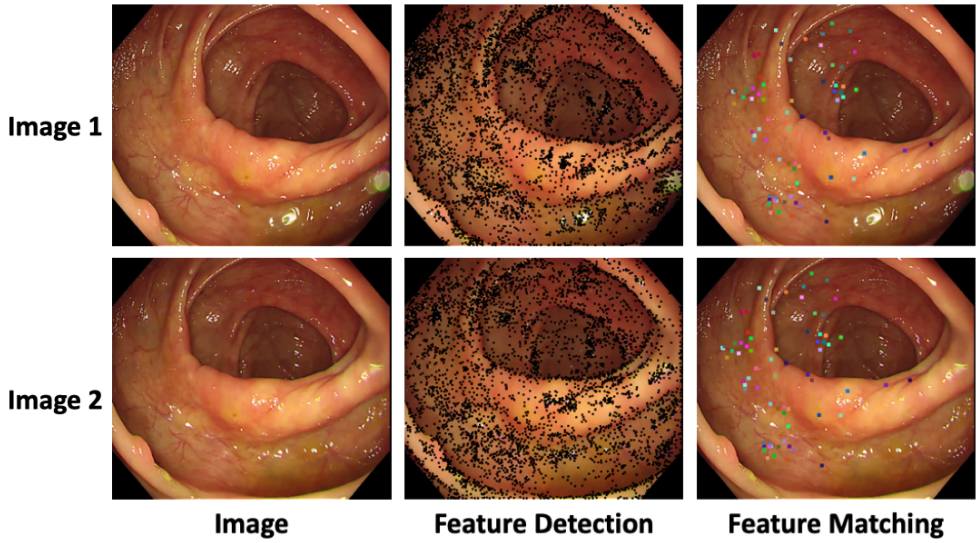


Fig. 2.1 This figure shows the Hessian affine features and their matches detected in two colonoscopy images. The matching features are plotted with the same colour.

2.2 Structure from Motion

Our method measures relative depth via structure from motion [36]. This section explains how structure from motion can estimate the 3D reconstruction of the scene from 2D images.

2.2.1 Feature Detection and Matching

The first step in structure from motion is to identify features that appear in more than one image. Then, the 3D coordinates of such features can be obtained via triangulation [10]. Different algorithms can be used to detect and match the features. In this project, Hessian affine feature detector [26], histogram of oriented gradients (HOG) feature descriptor [25], and nearest neighbour (NN) matching algorithms [28] are used.

Hessian affine feature detector [26] finds interest points based on the Hessian matrix evaluated at each pixel. For a pixel at $\mathbf{x} = (u, v)^T$, the scale-adapted Hessian matrix is defined as following.

$$\mu(\mathbf{x}, \sigma_I, \sigma_D) = \sigma_D^2 g(\sigma_I) \otimes \begin{bmatrix} L_{uu}(\mathbf{x}, \sigma_D) & L_{uv}(\mathbf{x}, \sigma_D) \\ L_{vu}(\mathbf{x}, \sigma_D) & L_{vv}(\mathbf{x}, \sigma_D) \end{bmatrix} \tag{2.1}$$

In Equation 2.1, $L(\mathbf{x})$ represents the image signal smoothed with a Gaussian kernel $g(\sigma_D)$, where σ_D is defined as the derivation scale. The Hessian of the smoothed signal, which

describes the gradient in each direction, is averaged across the neighbourhood of the pixel, using another Gaussian kernel $g(\sigma_I)$, where σ_I is defined as the integration scale. Then, the local extrema of the determinant of μ are selected as features.

The detected features are described using HOG descriptor [25]. In this algorithm, the image is divided into cells and the oriented gradients of the pixels within the cell are binned into a histogram. To achieve invariance under brightness and contrast, the histograms are normalised based on the local intensity.

The features in different images are then matched using NN matching [28]. The distance to the nearest neighbour and that to the second nearest neighbour is compared to filter out ambiguous matches (i.e. Lowe's ratio filtering [22]). Figure 2.1 shows the result of feature detection and matching.

2.2.2 Incremental Reconstruction

The next step is to estimate the projection matrices of the images and the 3D feature coordinates, based on the feature correspondences identified in the first step. More formally, given the pixel coordinates $\mathbf{x}_{ij}$ of the i-th feature in j-th image, the objective is to find the world coordinates $\mathbf{X}_i$ and the projection matrices $\mathbf{P}_j$ that minimise the reprojection error [11].

$$\mathbf{X}_i, \mathbf{P}_j = \underset{\mathbf{X}_i^*, \mathbf{P}_j^*}{arg\min} \sum_{i,j} v_{ij} \, f_d(\mathbf{P}_j^* \mathbf{X}_i^*, \mathbf{x}_{ij})^2 \tag{2.2}$$

In Equation 2.2, v_{ij} is 1 if the i-th feature appears in the j-th image and is zero otherwise. The function $f_d(\mathbf{P}_j^* \mathbf{X}_i^*, \mathbf{x}_{ij})$ measures the distance between the projection of the estimated 3D feature coordinates and its 2D coordinates. This optimization problem can be solved incrementally, starting from a pair of images [12]. The incremental reconstruction algorithm consists of the following three steps.

- Firstly, the two images with enough camera motion (i.e. large distance between camera centres) is selected to be the initial pair. This pair is selected as following: For every pair of images, rotation-only camera model is applied to align the two images. The number of features that cannot be explained via rotation-only model is identified as outliers, and the image pair with the highest number of outliers is selected as the initial pair. This step ensures that the baseline connecting the two camera centres is long enough to allow accurate triangulation.

- Secondly, the reconstruction between the initial pair is obtained. This is done by estimating the essential matrix ($\mathbf{E} = \mathbf{T}_\times \mathbf{R}$ where $\mathbf{T}_\times$ and $\mathbf{R}$ represent the relative

translation and rotation between the two camera centered coordinates) from the feature correspondences. Such estimation can be done via different algorithms including the five-point method [19] and the eight-point method [9]. Then, the estimated $\mathbf{E}$ is decomposed into $\mathbf{T}_\times$ and $\mathbf{R}$ via singular value decomposition [31]. The reconstruction is then obtained via triangulation.

- Then, the remaining images are added to the reconstruction. When a new image is added, its camera pose is estimated in accordance with the current estimate for $\mathbf{X}_i$. Then, $\mathbf{X}_i$ are re-triangulated. Such iterative optimization can be performed via bundle adjustment [35].

2.2.3 Densification using Patch Match

Lastly, the sparse reconstruction (consisting only of features) is densified via PatchMatch [2]. The PatchMatch algorithm estimates the 3D coordinates and orientation of a plane (i.e. patch) corresponding to each pixel. In order to achieve this goal, the correspondence between pixels of different images should be identified. For computational efficiency this search is limited to be around the epipolar line. Then, the algorithm finds the patch coordinates and orientation that best explain its projection on different images.

2.3 Depth Estimation via Neural Network

The data generated from structure from motion is used to train a pixel-wise depth estimation network. This section introduces how neural networks can be used to estimate depth.

2.3.1 Convolutional Encoder-Decoder Architecture

Convolutional encoder-decoder architecture is widely used to predict per-pixel output for the given image. Variations of this architecture have shown accurate results for tasks such as image segmentation, instance segmentation and depth regression.

An example of a simple convolution encoder-decoder architecture is provided in Figure 2.2. Suppose an RGB image of width W and height H is given as an input. The role of the encoder is then to map this image into some latent representation. This is generally done by convolving the image with a set of filters, each representing a visual feature (e.g. edges and corners). The convolution layers are often followed by a pooling layer which down-samples the image into lower resolution.

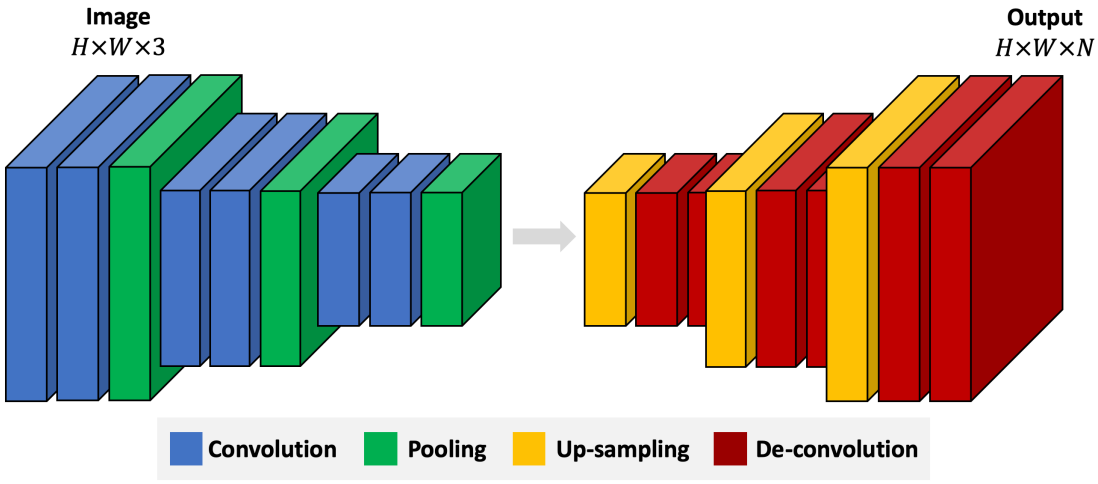


Fig. 2.2 This figure shows a simple convolutional encoder-decoder architecture.

The latent representation is then decoded via up-sampling and de-convolution, to produce the output of dimension $H \times W \times N$, where N is the number of output channels. For image segmentation, N is the number of classes each pixel can be assigned to. In this case, softmax activation is applied to the output layer so that each output can be interpreted as class probabilities.

The same architecture can be used to predict continuous variables, such as depth. In this case, the number of output channel is set to be 1 and a linear activation is applied.

2.3.2 Encoder Weight Initialization

Training an encoder-decoder network can be challenging with limited amount of data. A possible solution is to initialise the encoder with the weights pre-trained on a large-scale dataset (i.e. transfer learning [29]).

For example, the ImageNet dataset [33] can be used to pre-train the encoder weights. This dataset consists of 14 million hand-annotated images of more than 20,000 categories [33]. Since the classification task is very challenging, the encoder is forced to learn wide variety of features. If the pre-trained weights are good enough to produce a latent representation that is well-descriptive of the input image, the encoder weights can be fixed during training.

Chapter 3

Method

This chapter explains the three steps of our approach in detail. Firstly, sparse relative depth is estimated from monocular colonoscopy images. Secondly, the generated ground truth is used to train a depth estimation model. Lastly, the model's prediction is scale-matched to the ground truth to allow quantitative and qualitative evaluation. These steps are illustrated in Figure 3.1.

3.1 Data Generation

It is challenging to estimate the 3D reconstruction from monocular colonoscopy images due to non-rigid colon motion and lack of visual features. We overcome these challenges in two steps. Firstly, the colonoscopy images are filtered and divided into short sequences. Then, structure from motion is applied to each sequence separately to obtain its sparse 3D reconstruction and the corresponding depth.

3.1.1 Filtering

To ensure high accuracy of the reconstruction, we apply two steps of filtering on the images decoded from the raw colonoscopy videos. Firstly, the images taken with the plastic cap and those taken outside of the colon are discarded (see Figure 3.2 for a typical time-line of a colonoscopy video).

The images with plastic cap are discarded for the following reason. In these images, the feature matches are identified both on the colon surface and on the plastic cap. While the features on the colon surface move relative to the camera motion, the features on the cap move together with the camera. Such discrepancy in the feature motion results in inaccurate estimation of the camera pose and the 3D point cloud.

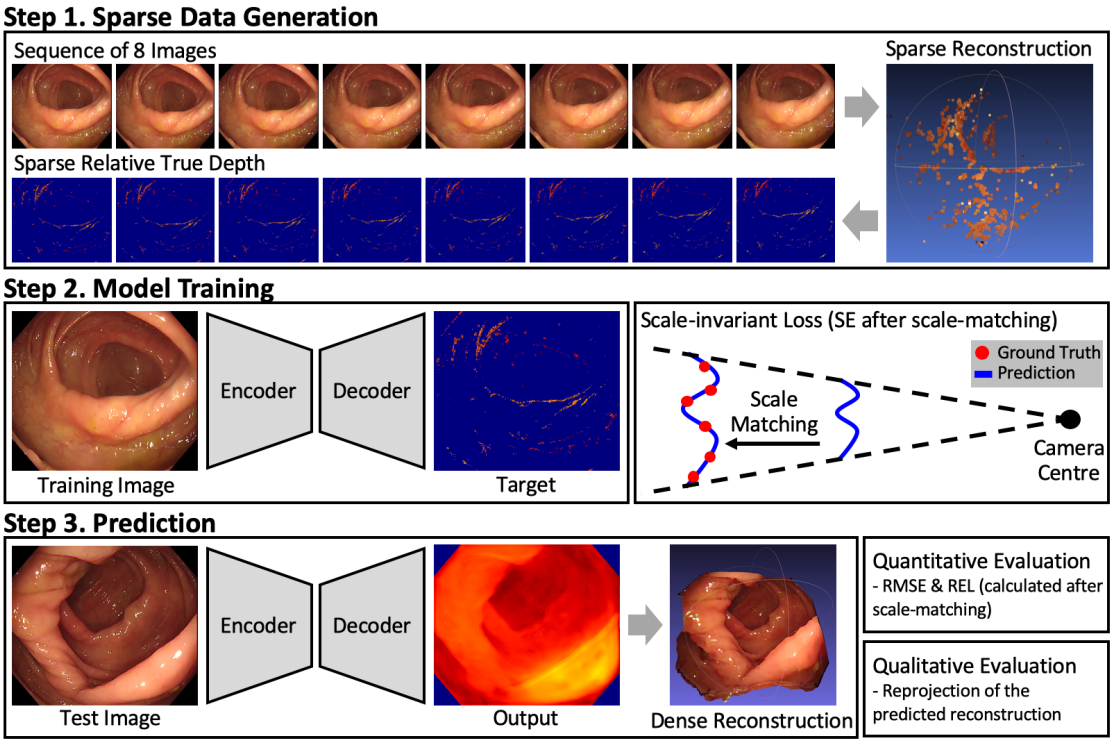


Fig. 3.1 This figure illustrates the three steps of our method. In the first step, structure from motion is applied to sequences of colonoscopy images to estimate their 3D reconstruction and the corresponding depth. Then, the generated ground truth is used to train a convolutional encoder-decoder network. Since the ground truth is in arbitrary scale, we use scale-invariant loss for training. Lastly, the predictions on unseen images are scale-matched to the corresponding ground truth to allow quantitative and qualitative evaluation.

For the remaining images (i.e. images taken inside the colon without the plastic cap), the interest points are detected and described using the Hessian affine detector [26] and HOG descriptor [25]. Figure 3.3 shows how the number of features changes during the colonoscopy procedure. Since structure from motion estimates the camera poses and the 3D feature coordinates from the feature correspondences, the images with small number of features generally result in inaccurate reconstruction. To ensure high accuracy of the ground truth, the images with less than 1000 features are discarded.

3.1.2 Sparse Structure from Motion

While the aforementioned filtering can help removing highly challenging images, it is still difficult to obtain dense reconstruction of the colon due to non-rigid motion and the lack of unique visual features. To overcome these challenges and obtain accurate 3D reconstruction,

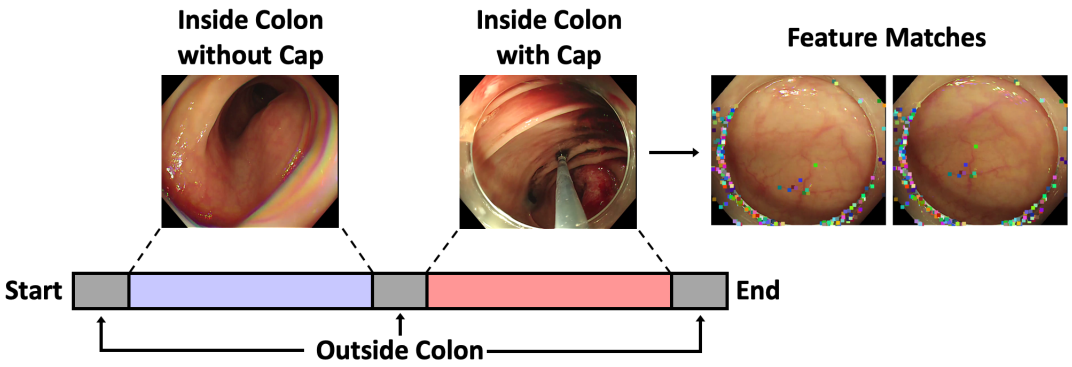


Fig. 3.2 This figure illustrates the time-line of a colonoscopy video. The video normally starts from the outside of the colon. Then, the physician inserts the endoscope into the patient's colon to start scanning for polyps. In some videos, the physician retrieves the endoscope, attaches a plastic cap at the end of the device, and inserts the device again. Such technique is called cap-assisted colonoscopy [27] and is used to improve the polyp detection rate. For the images taken with the plastic cap, majority of the feature matches are identified on the plastic cap.

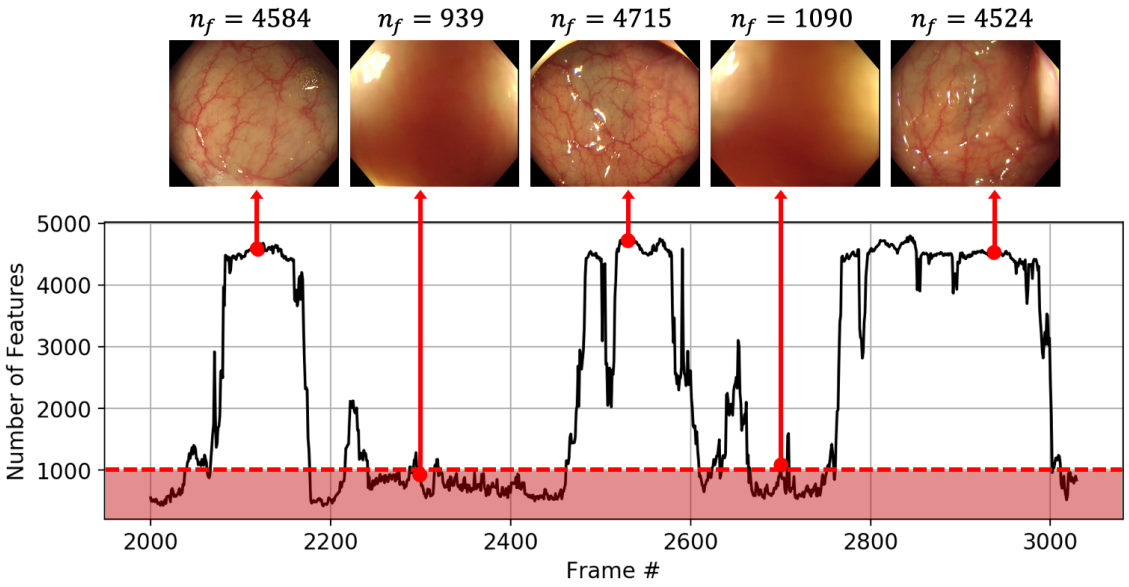


Fig. 3.3 This figure illustrates the change in the number of features during colonoscopy procedure. When the physician is scanning the surface for polyps, the camera motion is small and visual features such as the blood vessels are clearly visible. This results in high number of features. Once the scanning is complete, the physician quickly moves the camera to scan different part of the colon. Such quick motion creates huge motion blur, resulting in low number of features. Images with less than 1000 features are discarded to ensure high accuracy of the reconstruction.

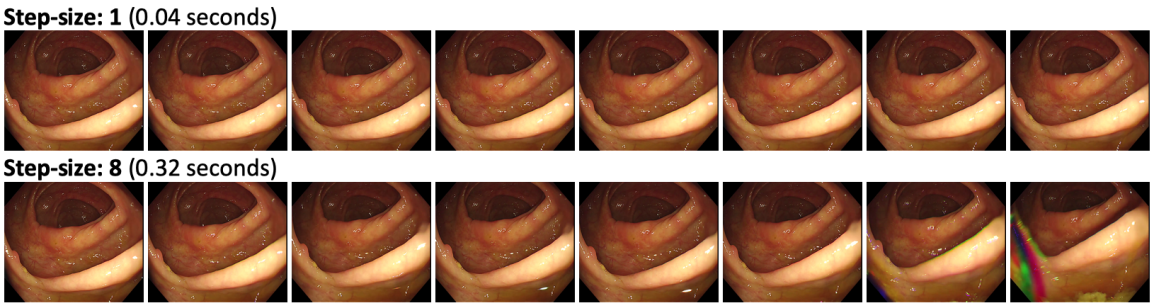


Fig. 3.4 This figure shows two image sequences of different step-sizes starting from the same image. If the step-size is too small, there is not enough camera motion to utilize the multi-view geometry. In extreme case where there is no camera motion, the features can have arbitrary depth, and still be projected accurately (i.e. degenerate solutions). On the other hand, if the step-size is too big, the first and the last image may not share common features. Furthermore, the quality of the reconstruction gets affected by the non-rigid colon motion. Hence, the step-size should be selected carefully for the given dataset.

we applied several constraints on the structure from motion pipeline to obtain sparse, yet accurate 3D reconstruction (see Section 2.2 for more details of the structure from motion).

- Firstly, the colonoscopy images (that survived filtering) are divided into sequences of 8 consecutive images. While it is difficult, if not impossible, to reconstruct the whole colon, it is possible to obtain a sparse reconstruction of the local colon surface by applying structure from motion on each sequence separately.

  Each sequence is formed by selecting every N-th image, where the value of N is defined as the step-size of the sequence (see Figure 3.4 for examples). The step-size should be short enough to assume rigid surface but long enough to ensure sufficient camera motion.

- Secondly, the camera poses (and 3D feature coordinates) are estimated from the feature correspondences. In order to ensure accurate camera poses, only the features that appear in all 8 images are considered in the incremental reconstruction. Also, any sequence for which the algorithm failed to estimate all 8 camera poses is discarded. This filtering ensures that the reconstruction can be explained by all the images in the sequence.

- Lastly, the reconstruction is densified via PatchMatch [2] algorithm. This step uses the camera poses estimated from the previous step. A patch is reconstructed if it is consistent in more than four views.

For successfully reconstructed sequences, the 3D reconstruction is projected onto each image to yield sparse pixel-wise relative depth. In this project, the depth of a pixel is defined as the distance from the camera centre to the corresponding 3D coordinates, measured along the optical axis. Note that the reconstruction obtained by the structure from motion is in arbitrary scale and so is the depth.

3.2 Model Training

The generated ground truth is used to train a convolutional encoder-decoder network with single output channel. A typical semantic segmentation architecture can be used. Since the ground truth is sparse and has arbitrary scale, a suitable loss function is required to train the model. This is achieved in the following steps.

- Firstly, the ground truth depth of each image (which has an arbitrary scale) is normalised to have the same average of 100.

$$\bar{d}^{\text{true}}_{u,v} = d^{\text{true}}_{u,v} \times \frac{N_{\text{true}}}{\sum_{u',v'} d^{\text{true}}_{u',v'}} \times 100 \tag{3.1}$$

  In Equation 3.1, $d^{\text{true}}_{u,v}$ and $\bar{d}^{\text{true}}_{u,v}$ represent the true depth and the normalised true depth at pixel (u,v). N_{true} represents the total number of pixels with ground truth.

- When a model makes prediction for a training image, the scale of the prediction is matched to that of the normalised true depth. This is done by multiplying the prediction by a factor s that minimises the squared distance from the normalised true depth.

$$s = \underset{s^*}{arg\min} \sum_{u,v} \mathbb{I}\left(d^{\text{true}}_{u,v} > 0\right) \left(\bar{d}^{\text{true}}_{u,v} - s^* \times d^{\text{pred}}_{u,v}\right)^2 \tag{3.2}$$

  Equation 3.2 shows how the value of s can be calculated. $d^{\text{pred}}_{u,v}$ represents the predicted depth at pixel (u,v). The value of $\mathbb{I}\left(d^{\text{true}}_{u,v} > 0\right)$ is 1 if there is true depth assigned for that pixel, and 0 otherwise.

- After the scale is matched, the squared error at each pixel is calculated and back-propagated to update the model weights.

$$e_{u,v} = \mathbb{I}\left(d_{u,v}^{\text{true}} > 0\right)\left(\bar{d}_{u,v}^{\text{true}} - s \times d_{u,v}^{\text{pred}}\right)^2 \tag{3.3}$$

Equation 3.3 shows the mathematical expression of the training loss at pixel (u, v). Note that the loss is zero for the pixels with no ground truth. Therefore, only the model weights that are responsible for the pixels with true depth are updated at each batch.

The first step may seem unnecessary since the prediction can be matched to the true depth of any scale using Equation 3.2. However, if this step is omitted, the training loss calculated after the scale-matching becomes proportional to the scale of the true depth. As a result, the model training becomes biased to the sequences with large scale.

Another advantage gained by the first step is that it can limit the search space for the scale factor s. To reduce the computational cost, the appropriate scale factor is selected from a discrete set of values. Since the true depth is already normalised to have the same average, the search can be done only on a small number of choices, allowing faster training.

3.3 Prediction

Since the model is trained with the scale-invariant loss, its output should be interpreted as the relative depth. Therefore, the model's prediction on the validation (and test) images should also be scale-matched to the corresponding ground truth. Equation 3.2 is used again to find the suitable scaling factor.

Once the scale of the prediction is adjusted, the accuracy of the prediction can be evaluated both quantitatively and qualitatively. See Section 4.3 for detailed information on the evaluation protocol (e.g. accuracy metrics).

Chapter 4

Experiment Setup

This chapter provides the implementation details of our method. The first and second section explain how the data generation and network training are implemented. The last section introduces the evaluation protocol that was used to measure and compare the performance of different depth estimation models.

4.1 Data Generation

This section introduces the internal dataset and explains the implementation of the sparse structure from motion.

4.1.1 Dataset

23 videos, each containing a full colonoscopy procedure, are used as the raw data. The videos are recorded in `mp4` format, and every video has the resolution of 1250×1080 and frame-rate of 25 frames per second. The videos are decoded into `png` images using FFmpeg [8]. Table 4.1 shows the change in the number of frames in each video during filtering. Examples of the images that survived the filtering are shown in Figure 4.1.

4.1.2 Sparse Structure from Motion

The structure from motion pipeline is implemented using a Python package OpenSfM [24]. Following set of parameters are used (default values are used for other parameters).

- `feature_type = HAHOG`. This parameter specifies the feature detector and descriptor. `HAHOG` stands for the Hessian affine detector and HOG descriptor.

		Number of Frames		
Video ID	Duration	Raw	Inside Colon & Without Cover	Featureless Filtered
1	30m 21s	45,528	18,501	11,244
2	9m 50s	14,760	13,764	10,593
3	8m 19s	12,480	10,838	9,618
4	28m 5s	57,144	56,977	45,101
5	15m 53s	23,832	23,764	17,712
6	19m 16s	28,908	9,438	5,763
7	11m 56s	17,904	11,841	8,389
8	25m 50s	38,772	38,324	30,869
9	26m 11s	39,288	39,288	19,001
10	13m 55s	20,880	19,514	16,134
11	22m 23s	33,588	30,652	25,657
12	18m 7s	27,192	26,952	19,725
13	26m 11s	39,288	39,288	34,924
14	21m 17s	31,945	31,470	20,207
15	33m 8s	49,730	8,947	5,352
16	16m 36s	24,913	24,434	21,325
17	28m 53s	43,346	5,085	3,430
18	32m 56s	49,406	49,185	37,623
19	36m 7s	54,194	52,621	46,701
20	14m 16s	21,409	17,854	12,463
21	46m 7s	69,182	11,033	8,362
22	42m 15s	63,410	62,701	55,082
23	25m 08s	37,717	37,133	34,561
Total	9h 13m	995,768	639,604	499,836

Table 4.1 Change in the number of frames during data filtering

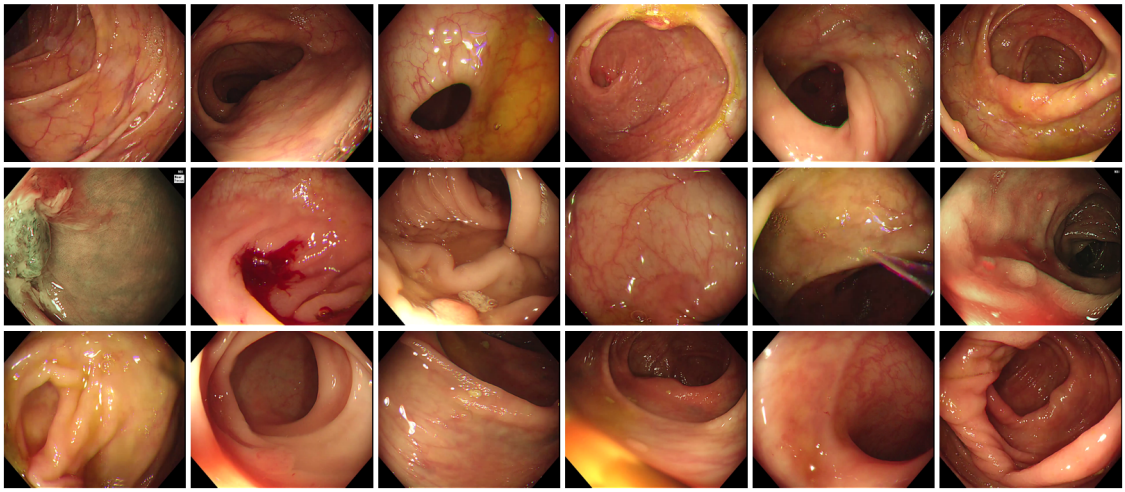


Fig. 4.1 This figure shows examples of colonoscopy images in the dataset that survived the image filtering.

- `matcher_type = FLANN`. This parameter specifies the algorithm to use for the feature matching. FLANN(Fast Library for Approximate Nearest Neighbours)[28] is used with the Lowe's ratio of 0.8.
- `min_track_length = 8`. This parameter specifies the minimum length of the feature tracks. By setting this parameter to 8, we only consider the features that appear in all the images.
- `bundle_outlier_fixed_threshold = 0.006`. This parameter specifies the threshold for outlier filtering during bundle adjustment. If the reprojection error of a feature divided by the image width is larger than 0.006, the feature is discarded (for our dataset, this corresponds to 7.5 pixels). This threshold filters out any non-rigidly moving features and thus improves the quality of the reconstruction.
- `depthmap_min_consistent_views = 4`. This parameter specifies the number of consistent views required to reconstruct a patch during PatchMatch [2]. As the value of this parameter decreases, more patches can be reconstructed, but the quality of the reconstruction becomes poorer. The value of 4 is selected empirically.

Using the aforementioned parameters, the structure from motion pipeline is applied to sequences of 8 consecutive colonoscopy images. In order to select the step-size suitable for the dataset, five videos are divided into sequences of four different step-sizes - 1, 2, 4 and 8. Then, the structure from motion is applied to each step-size to measure the survival rate (i.e. rate of successful reconstruction). For successfully reconstructed sequences, the camera

Step-size	Survival Rate	Average Camera Motion
1	0.203	19.09
2	0.206	26.22
4	0.152	26.01
8	0.075	32.73

Table 4.2 Survival rate and camera motion measured for different step-sizes

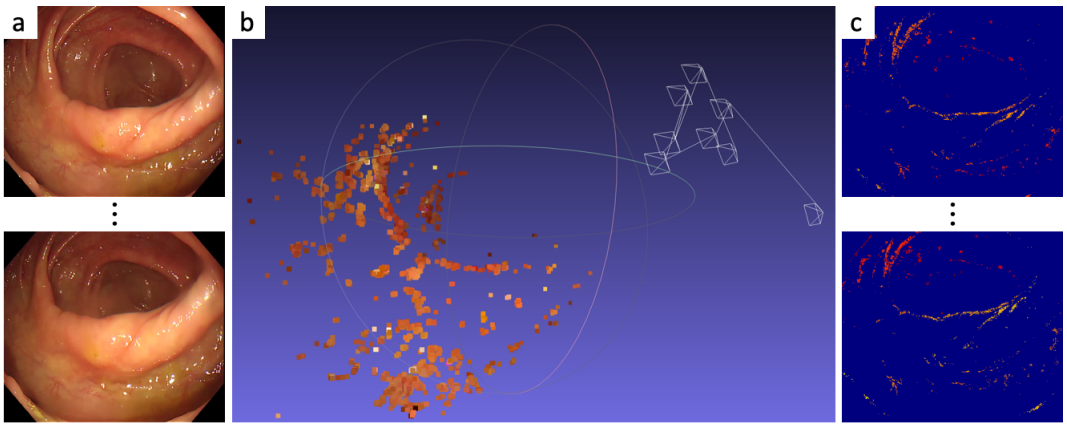


Fig. 4.2 Figure (a) shows monocular colonoscopy images. Figure (b) shows the sparse reconstruction and camera poses estimated from these images. Figure (c) shows the pixel-wise relative depth obtained by projecting the reconstruction to each image.

motion is measured. Camera motion is defined as the distance between the furthest camera pairs. Since each reconstruction has different scale, the camera motion is divided by the average distance from the cameras to the reconstruction and is multiplied by 100.

The obtained survival rate and camera motion are shown in Table 4.2. The survival rate sharply decreases for step-size larger than 2. This can be due to non-rigid colon motion or the absence of feature correspondences between the first and the last frame. For our dataset, the survival rate is highest for the step-size of 2.

For this step-size, the average camera motion is 26.22. This means that if the average distance from camera to the colon surface is $10cm$, the camera centre moves about $2.62cm$ within the sequence. Such change in view is sufficient to allow accurate triangulation. For this reason, step-size of 2 is applied to all the videos. Since the frame-rate of the videos is 25 frames per second, this means that the consecutive images in a sequence are separated by 0.08 seconds. As a result, $55,492$ sequences are generated.

Of the generated sequences, $8,480$ sequences are successfully reconstructed. For successfully reconstructed sequences, the 3D reconstruction is projected to each image to obtain

Dataset	# Videos	# Snippets
Training	17	5802
Validation	3	924
Test	3	1754

Table 4.3 Data separation

the pixel-wise depth. Figure 4.2 shows an example of a reconstruction and the corresponding depthmaps obtained from one of the sequences. The sequences are then split into three sets - training, validation and test. The sequences extracted from the same video (i.e. same patient) are assigned to the same set. Table 4.3 shows how the dataset is split.

To reduce the computational cost in model training and to improve the smoothness of the result, the images and the depthmaps are resized to 25% (50% in width and height). The resized images and depthmaps are then stored in `npy` format.

4.1.3 Data Quality Evaluation

It is important to check if the generated ground truth is correct. Since there is no reference true depth (e.g. measured from sensors) available for this dataset, the quality of the data is evaluated indirectly by measuring the following quantities.

- **Reprojection error.** The reprojection error of the i-th feature can be defined as following.

$$e_{\text{reproj},i} = \frac{1}{8}\sum_{j=1}^{8} f_d\left(\mathbf{P}_j\mathbf{X}_i, \mathbf{x}_{ij}\right) \tag{4.1}$$

  In this equation, $\mathbf{P}_j$ is the estimated projection matrix of the j-th image. $\mathbf{X}_i$ is the estimated 3D coordinates of the i-th feature. $\mathbf{x}_{ij}$ is the pixel coordinates of the i-th feature in the j-th image. The function f_d measures how far the projection is from the feature's pixel coordinates. If both the camera poses and the 3D feature coordinates are accurate, the reprojection error should be small.

- **Camera motion.** It is important to note that inaccurate reconstruction can still result in small reprojection error if there is small or no camera motion. In order to check if the sequence has sufficient camera motion, the distance between the furthest pair of

camera poses is measured. To account for arbitrary scale of the reconstruction, this distance is divided by the average depth and is multiplied by 100.

- **Feature motion.** Another way of confirming that there is sufficient camera motion is to measure the feature motion across the images. For every feature that appears in all 8 frames, its pixel coordinates in different frames are compared, and the distance between the furthest pair is defined as its motion.

If the reprojection error is small for a sequence, and there is sufficient camera and feature motion between images, the quality of the obtained ground truth can be trusted.

4.2 Network Training

In this project, we use three variants of the convolutional encoder-decoder network - U-Net [32], FPN [20], and LinkNet [4]. The models are designed to have a single output channel with linear activation, so that the output can be interpreted as depth. For all models, ResNet [13] with 34 layers is used as the backbone (i.e. feature extractor). The models are built using a Python library SegmentationModels [37] which is built on Keras [6].

At the beginning of each epoch, the images in the training set are shuffled and divided into batches of 8. The model is trained for 30 epochs, using Adam optimizer [18]. The learning rate is set to be 0.001 and is halved if the validation loss does not improve for five epochs.

The scale-matching function is implemented as following: Given a normalised ground truth and a prediction, the prediction is multiplied by a set of discrete numbers, ranging from 0.5 to 2.0, incremented by 0.1. The value that minimises the squared error is selected. If this value is s, the next decimal is identified by trying values from $s-0.09$ to $s+0.09$, incremented by 0.01. Such process is repeated to find the optimal scale factor up to the fourth decimal.

4.3 Performance Evaluation

After training the depth estimation model with the ground truth generated from the sparse structure from motion, the model's performance is evaluated on validation (and test) images. This section introduces the evaluation protocol that is used to evaluate and compare the performance of different models.

4.3.1 Sparse Evaluation

The prediction error can be measured quantitatively for the pixels with ground truth. We use the following accuracy metrics to measure the prediction error.

- **Root mean square error (RMSE).** This metric measures the root mean square error between the prediction and the normalised ground truth, after scale-matching. Since the normalised true depth is a dimensionless quantity, this metric does not provide any information on how big the error is in physical unit (e.g. in *mm*). Note that this metric can still be used to compare the relative performance between different models.

$$\text{RMSE} = \sqrt{\frac{1}{N_{\text{truth}}} \sum_{u,v} \mathbb{I}\left(d_{u,v}^{\text{true}} > 0\right) \left(\bar{d}_{u,v}^{\text{true}} - s \times d_{u,v}^{\text{pred}}\right)^2} \tag{4.2}$$

- **Average relative error (REL).** This metric measures the average relative error (prediction error divided by the true depth), after scale-matching.

$$\text{REL} = \frac{1}{N_{\text{truth}}} \sum_{u,v} \mathbb{I}\left(d_{u,v}^{\text{true}} > 0\right) \frac{|\bar{d}_{u,v}^{\text{true}} - s \times d_{u,v}^{\text{pred}}|}{\bar{d}_{u,v}^{\text{true}}} \tag{4.3}$$

- **Absolute root mean square error (aRMSE).** This metric measures the root mean square error in absolute scale (e.g. in *mm*). This is obtained by multiplying the RMSE by some conversion coefficient c (see Section 4.3.3 for detail on the estimation of c).

$$\text{aRMSE} = c \times \text{RMSE} \tag{4.4}$$

4.3.2 Dense Evaluation

The aforementioned metrics - REL, RMSE and aRMSE - can only measure the accuracy at the pixels with ground truth, the number of which is generally only 2% of the total number of pixels. To evaluate the quality of the prediction as a whole, we use the following method.

- For a selected image sequence, depth is predicted for the first image. The prediction made at each pixel, together with its RGB values, can specify a 3D point with colour. By repeating this for all the pixels, a dense 3D point cloud corresponding to the prediction is obtained.

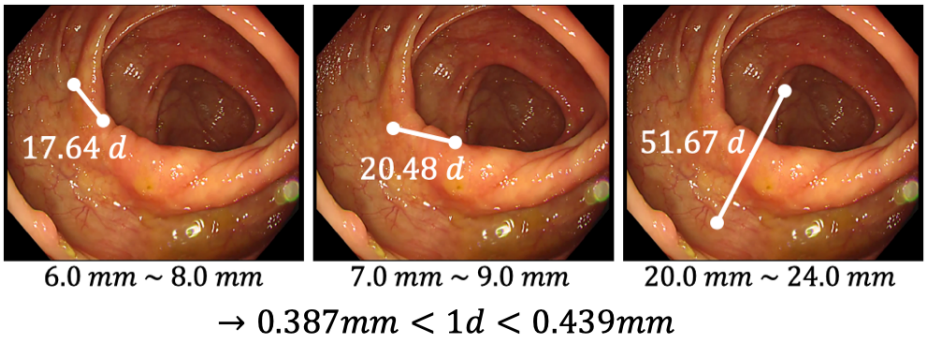


Fig. 4.3 This figure illustrates the method for estimating the true scale of the ground truth.

- The point cloud (i.e. predicted 3D reconstruction) is then projected on the remaining images. The camera poses estimated from the structure from motion is used to calculate the relative camera pose between images.
- The projection of the point cloud is compared to the image. If the prediction is accurate, the estimated 3D reconstruction would be an accurate representation of the local colon surface. Therefore, its projection on a different image should be similar to that image. The RGB difference between the two is used as a qualitative measure of the dense prediction.

4.3.3 True Scale Estimation

In order to calculate the error in absolute scale (i.e. aRMSE), the conversion coefficient should be estimated. This is done in the following steps (see Figure 4.3):

- For each sequence, N feature pairs are selected and their distances are measured in the normalised scale. Suppose that this value has a dimensionless unit d.
- For each feature pair, we make a conservative guess on the range of values its distance can have (e.g. in mm). This guess is made based on the fact that the diameter of the colon is generally 5 to 6cm when inflated.
- Then, it is possible to find the range of values the conversion coefficient can have in order to satisfy all the guesses. The mid-point of this range is selected as the conversion coefficient, and is multiplied to the RMSE measured on the sequence. This process is repeated for a subset of test sequences to obtain the aRMSE.

Chapter 5

Results

This chapter presents the key experiments of our project and their results. Firstly, the quality of the generated ground truth is evaluated. Secondly, the depth estimation models of different configuration are compared to identify the best-performing configuration. Lastly, the performance of the selected model is evaluated on the test set.

5.1 Quality of the Generated Ground Truth

In order to train and test the depth estimation model with the generated ground truth, it is important to check whether the obtained reconstruction is accurate.

The quality of the estimated camera poses and 3D feature coordinates can be evaluated by measuring three quantities - reprojection error, camera motion, and feature motion. Reprojection error measures how close the projections of the estimated 3D feature coordinates are to the corresponding pixel coordinates identified during feature detection. Camera motion is defined as the distance between the furthest camera pairs. Since each reconstruction has different scale, this value is divided by the average depth and is multiplied by 100 for normalisation. Lastly, feature motion is defined as the change in the feature's pixel coordinates within the sequence (see Section 4.1.3 for more detail).

Figure 5.1 shows the distributions of the three quantities. The reprojection error is 3.32 pixels in average, and is less than 4.24 pixels for 90% of the sequences. This suggests that the estimated 3D feature coordinates and the camera poses can accurately explain the pixel coordinates of the features in each image. However, it should be noted that the reprojection error can be small for an inaccurate reconstruction if there is small or no camera motion. In extreme case where the 8 images are identical, the features can have arbitrary depth.

We confirm that most of the sequences in our dataset have sufficient motion. The camera motion is 22.86 in average and is larger than 9.66 for 90% of the sequences. Considering

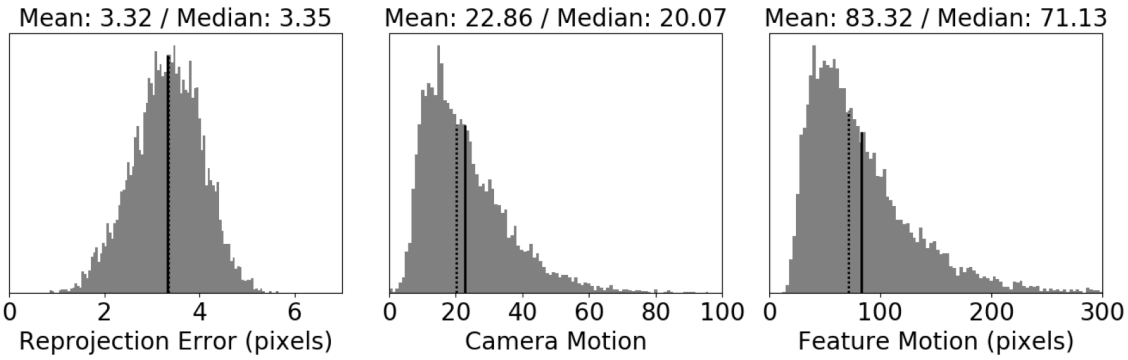


Fig. 5.1 These plots show the distribution of the reprojection error, camera motion, and feature motion. In each plot, the mean and the median are marked with a solid line and a dashed line, respectively.

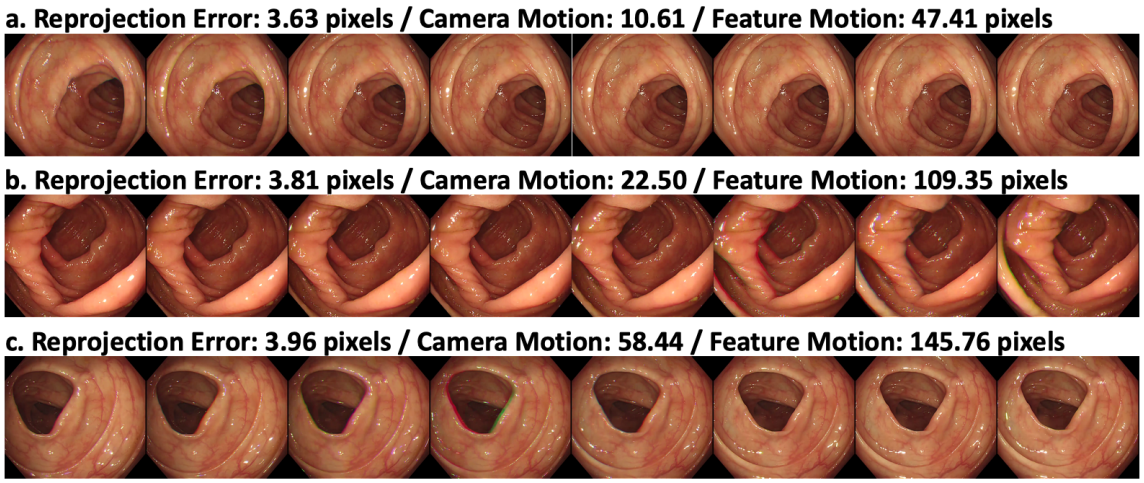


Fig. 5.2 This figure shows three sequences of different amount of camera motion. Large camera motion ensures that the obtained reconstruction is not a degenerate solution.

that the average depth is normalised to 100, this camera motion is sufficient to create enough parallax (see Figure 5.2 for examples). Another evidence for sufficient motion is that the feature motion is high (83.32 pixels in average and larger than 35.34 pixels for 90% of the sequences). The fact that small reprojection error is achieved for sequences of sufficient camera motion suggests that the obtained ground truth is accurate.

5.2 Model Selection

In this section, depth estimation models of different configuration are compared based on their performance on the validation set. Three experiments are conducted by varying the training loss, model architecture, and encoder weight initialization.

5.2.1 Training Loss Comparison

This experiment compares three models trained with different training losses. Following are the definition of the three loss functions. Note that $d_{u,v}^{\text{true}}$, $\bar{d}_{u,v}^{\text{true}}$ and $d_{u,v}^{\text{pred}}$ represent the ground truth, normalised ground truth and prediction at pixel (u,v).

- **Squared error (SE).** This training loss minimises the squared error with respect to the unnormalised ground truth. SE at pixel (u,v) can be defined as following.

$$\text{SE}_{u,v} = \mathbb{I}\left(d_{u,v}^{\text{true}} > 0\right)\left(d_{u,v}^{\text{true}} - d_{u,v}^{\text{pred}}\right)^2 \tag{5.1}$$

- **SE with respect to the normalised ground truth (NSE).** This training loss minimises the squared error with respect to the ground truth that is normalised to have the same average depth. NSE at pixel (u,v) can be expressed as following.

$$\text{NSE}_{u,v} = \mathbb{I}\left(d_{u,v}^{\text{true}} > 0\right)\left(\bar{d}_{u,v}^{\text{true}} - d_{u,v}^{\text{pred}}\right)^2 \tag{5.2}$$

- **NSE calculated after scale-matching (SNSE).** This training loss is similar to NSE except that the prediction is multiplied by a factor s to match the scale of the normalised ground truth (see Section 3.2 for more detail on scale-matching).

$$\text{SNSE}_{u,v} = \mathbb{I}\left(d_{u,v}^{\text{true}} > 0\right)\left(\bar{d}_{u,v}^{\text{true}} - s \times d_{u,v}^{\text{pred}}\right)^2 \tag{5.3}$$

All three models have the U-Net [32] architecture with ResNet34 [13] backbone (see Section 4.2 for other details regarding the network training). Table 5.1 compares the validation set accuracy of the three models. For visual comparison, the predictions on three validation images are provided in Figure 5.3.

The model trained with a simple squared error (SE) shows poor performance (RMSE of 38.00 and REL of 0.1415). This is because the model tries to fit to the ground truth that has an arbitrary scale. As a result, the model fails to learn the relative depth and produces flat prediction. The accuracy improves significantly if SE is calculated after normalising the ground truth (RMSE decreases by 27.56 and REL decreases by 0.0757). Since the ground truth of any image now has the same average, the model can learn how further away a pixel is compared to the average depth.

	Validation Set Accuracy	
Training Loss	RMSE	REL
SE	38.00	0.1415
NSE	10.44	0.0658
SNSE	9.07	0.0551

Table 5.1 Comparison between different training losses

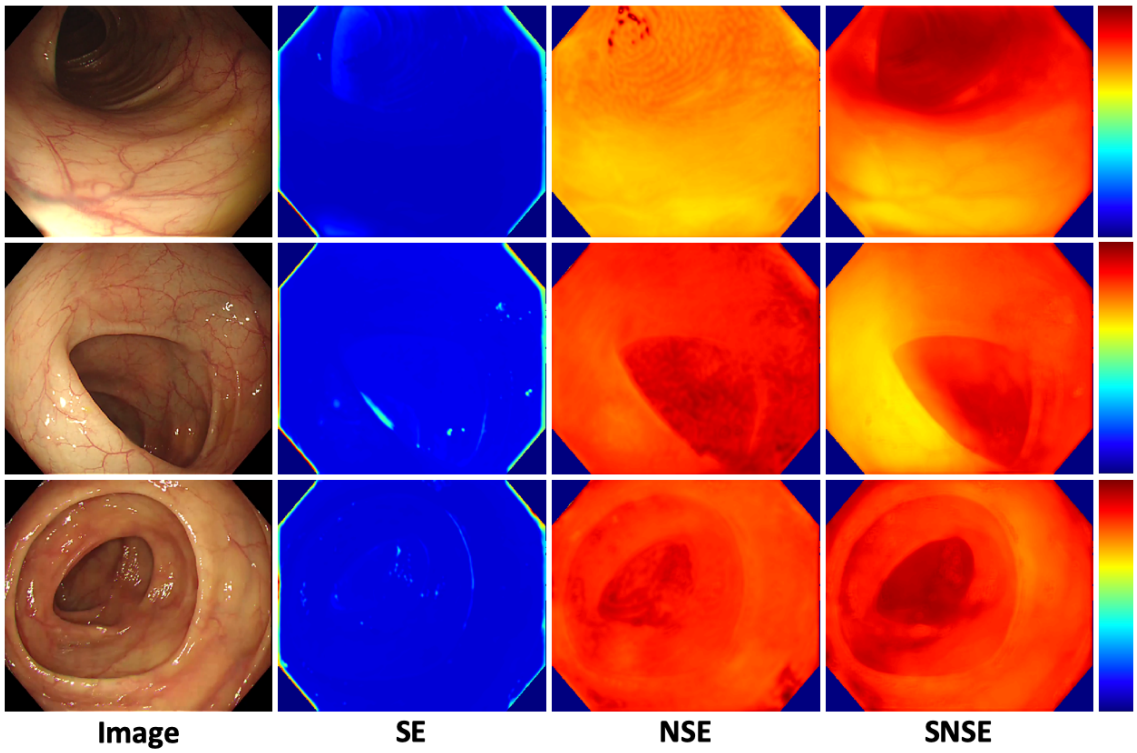


Fig. 5.3 This figure shows three validation images and the corresponding prediction made by the three models trained with different training losses. The model trained with SE produces nearly flat prediction. The model trained with NSE captures the differences in depth to some extent. However, the prediction is not smooth and the depth discontinuity at the circular folds is not clearly visible. On the contrary, the model trained with our scale-invariant loss produces smooth prediction with clearly visible circular folds.

However, there can be discrepancy between the scale of the ground truth even after the depth normalisation. The average depth of a sparse depthmap depends heavily on the distribution of the reconstructed points. For example, two consecutive frames should share similar scale. However, since they are assigned to a different sequence (note that the step-size is 2), and since the distribution of the reconstructed points is different between sequences, their average depth can differ by a significant amount. Our scale-invariant loss (SNSE) deals

	Validation Set Accuracy	
Model Architecture	RMSE	REL
U-Net	9.07	0.0551
FPN	9.11	0.0561
LinkNet	8.77	0.0535

Table 5.2 Comparison between different model architecture

with such inconsistency in scale by finding the scale that minimises the squared distance from the normalised ground truth. Since the loss is calculated after the scale-matching, the model is able to learn the relative depth between the pixels. As a result, the accuracy measured on the pixels with ground truth improves (RMSE of 9.07 and REL of 0.0551), and so does the quality of the dense prediction.

5.2.2 Model Architecture Comparison

This experiment compares three models of different architectures - U-Net [32], FPN [20] and LinkNet [4]. All three models have the ResNet34 [13] backbone. Based on the result of the previous experiment, the models are trained with the scale-invariant loss (SNSE). The validation set accuracy of the three models is provided in Table 5.2. Figure 5.4 shows the dense predictions on validation images.

All three models show small RMSE (ranging from 8.77 to 9.11) and REL (ranging from 0.0535 to 0.0561). While the accuracy measured on sparse points with ground truth is high for all models, the quality of the dense prediction is poor for FPN and LinkNet. The predictions made by these models fail to preserve the structural characteristics of the surface, such as the circular folds (see Figure 5.4).

On the contrary, the structure of the surface is clearly visible in the predictions made by U-Net. In U-Net, the input for each decoder block is appended by the feature maps produced in the corresponding encoder block. These skip connections help the model to maintain the structure of the input such as the edge locations, as can be seen in Figure 5.4. Such attribute is useful especially in colonoscopy depth estimation where the locations of the circular folds mark discontinuity in depth.

5.2.3 Encoder Weight Initialization Comparison

This experiment compares three different ways of initialising the encoder weights. The first model is randomly initialised. The second model is initialised with the weights pre-trained

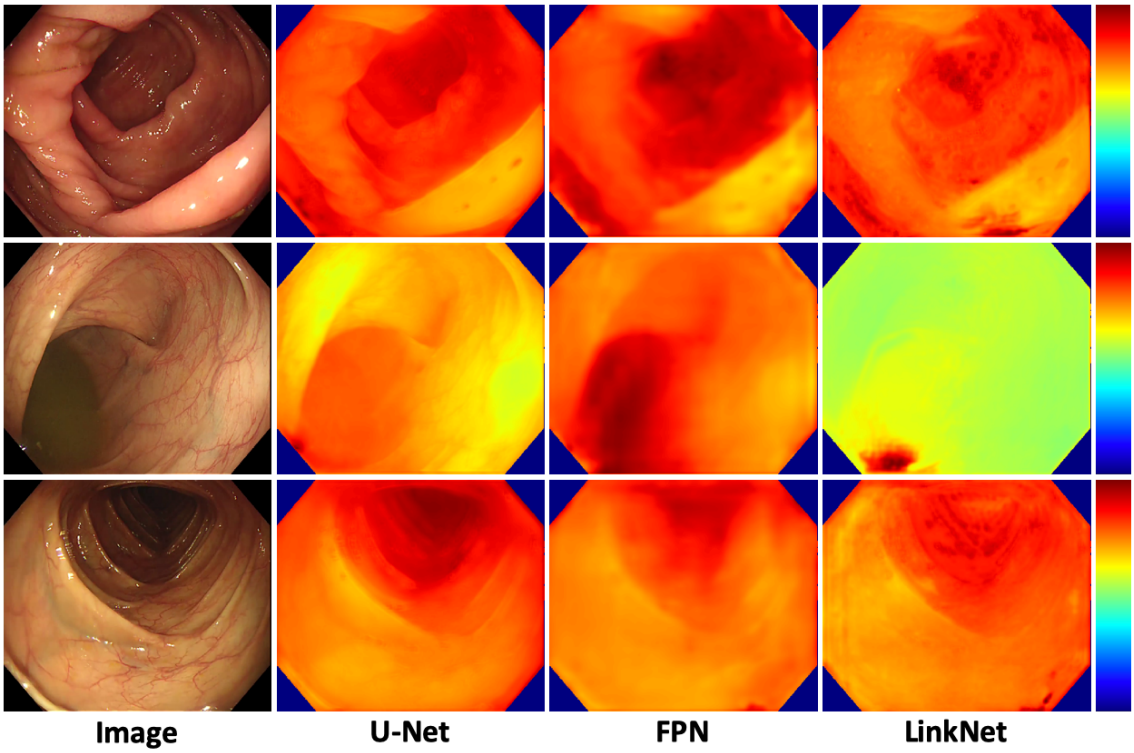


Fig. 5.4 This figure shows three validation images and the corresponding prediction made by the three models of different architecture. The prediction made by FPN and LinkNet is generally not smooth and does not show clear discontinuity at the circular folds, unlike those made by U-Net.

on the ImageNet dataset [33]. The third model is also initialised with the ImageNet weights but the weights are kept fixed during training (i.e. only the decoder weights are updated). All three models have the U-Net architecture with ResNet34 backbone and are trained with the scale-invariant loss. The validation set accuracy and the examples of dense prediction are provided in Table 5.3 and Figure 5.5, respectively.

While all three models achieve small RMSE (ranging from 8.72 to 9.33) and REL (ranging from 0.0535 to 0.0577), the quality of the dense prediction is poor for the models initialised with the ImageNet weights (both fixed and not fixed). This suggests that many of the features learned from a large-scale image dataset are not relevant for colonoscopy images. In such case, the latent representation produced by the encoder blocks becomes less informative of the input.

On the other hand, randomly initialised model learns from scratch the visual features highly relevant to the colonoscopy images. For example, the density of the blood vessels is a useful depth cue (the blood vessels look sparse if they are far from the camera). Learning

	Validation Set Accuracy	
Encoder Weight Initialization	RMSE	REL
Random Initialization	9.07	0.0551
ImageNet Weights	8.72	0.0535
ImageNet Weights + Fixed	9.33	0.0577

Table 5.3 Comparison between different encoder weight initialization

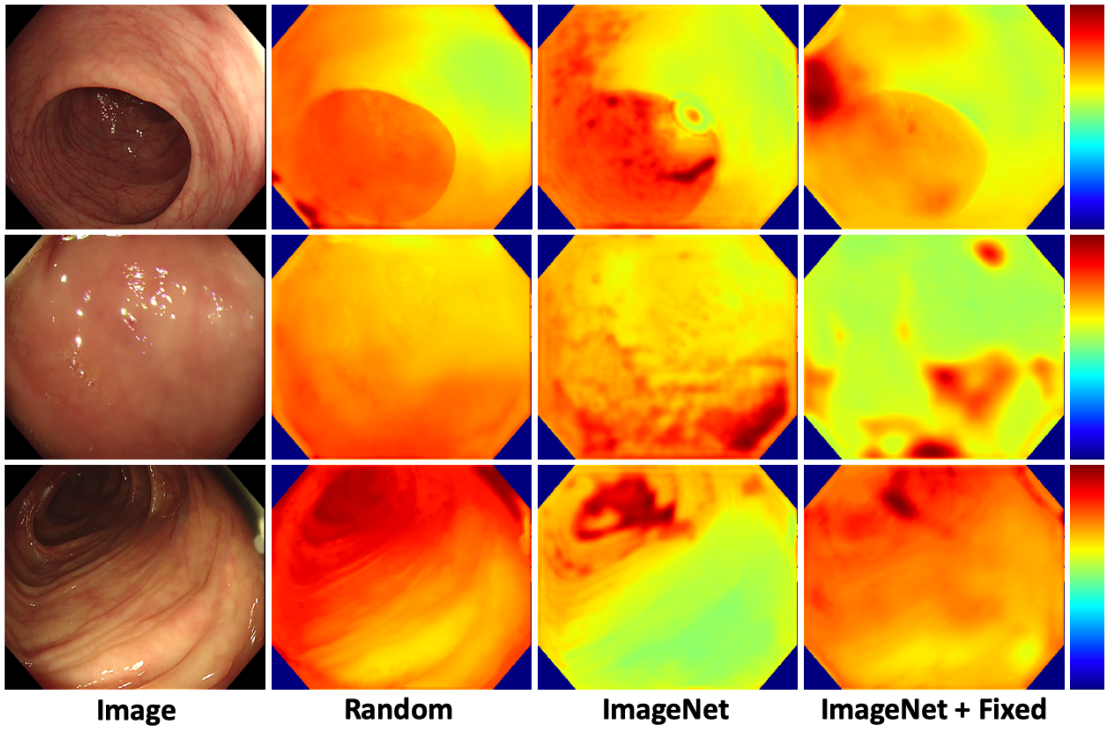


Fig. 5.5 This figure shows three validation images and the corresponding prediction made by the three differently initialised models. Initialising the encoder with the ImageNet weights leads to poor quality of the dense prediction (for both fixed and not fixed initialisation).

the features that can encode such information can therefore improve the quality of the depth prediction.

5.3 Performance Evaluation

The best-performing model configuration identified from the aforementioned experiments is (1) U-Net architecture with (2) random encoder weight initialisation, trained with the (3) scale-invariant loss. This model is evaluated on the test set, using the protocols introduced in Section 4.3.

Test Set Accuracy		
RMSE	REL	aRMSE
9.99	0.0566	2.7*mm*

Table 5.4 Test set accuracy of the best-performing model

5.3.1 Sparse Evaluation

For the pixels with ground truth, the prediction error can be measured quantitatively in terms of RMSE and REL. By multiplying RMSE with a suitable conversion coefficient, it is possible to obtain aRMSE, which is the RMSE measured in true scale (e.g. in *mm*). This quantity is calculated on a subset of 20 test sequences (see Section 4.3.1 for detailed definition of the three metrics).

RMSE and REL measured on the test set are 9.99 and 0.0566, respectively (see Table 5.4). This accuracy is consistent with the validation set accuracy of the same model (RMSE of 9.07 and REL of 0.0551). The prediction error, in true scale, is approximately 2.7*mm*. This suggests that the 3D reconstruction of the local colon surface estimated by the model, after suitable scale-conversion, can be used to accurately measure the polyp size and shape. Accurate depth estimates can also be used to assist the device control.

5.3.2 Dense Evaluation

Since the accuracy metrics can be calculated only on the pixels with ground truth, it is important to check whether the dense prediction is accurate. Figure 5.6 shows examples of the 3D reconstruction predicted by the model. The predicted mesh is smooth and captures the 3D structure of the surface, such as the circular folds.

In order to check whether the dense prediction is accurate, the 3D reconstruction obtained for one of the images in a sequence can be projected to other images in the same sequence. Such projection is possible since their relative camera pose was estimated during the incremental reconstruction (see 4.3.2 for more detail on dense prediction evalution). Figure 5.7 shows a typical example of such projection. Small difference between the projection and the image suggests that the predicted dense 3D reconstruction is accurate.

While the model prediction is accurate for most of the images, the quality of the dense prediction can be poor for highly challenging images. For example, the model makes incorrect prediction if there is water jet or the polyp removal device visible in the image. Both water jet and the device move together with the camera. In such case, the features identified on the water jet or the device can lead to unsuccessful or inaccurate reconstruction

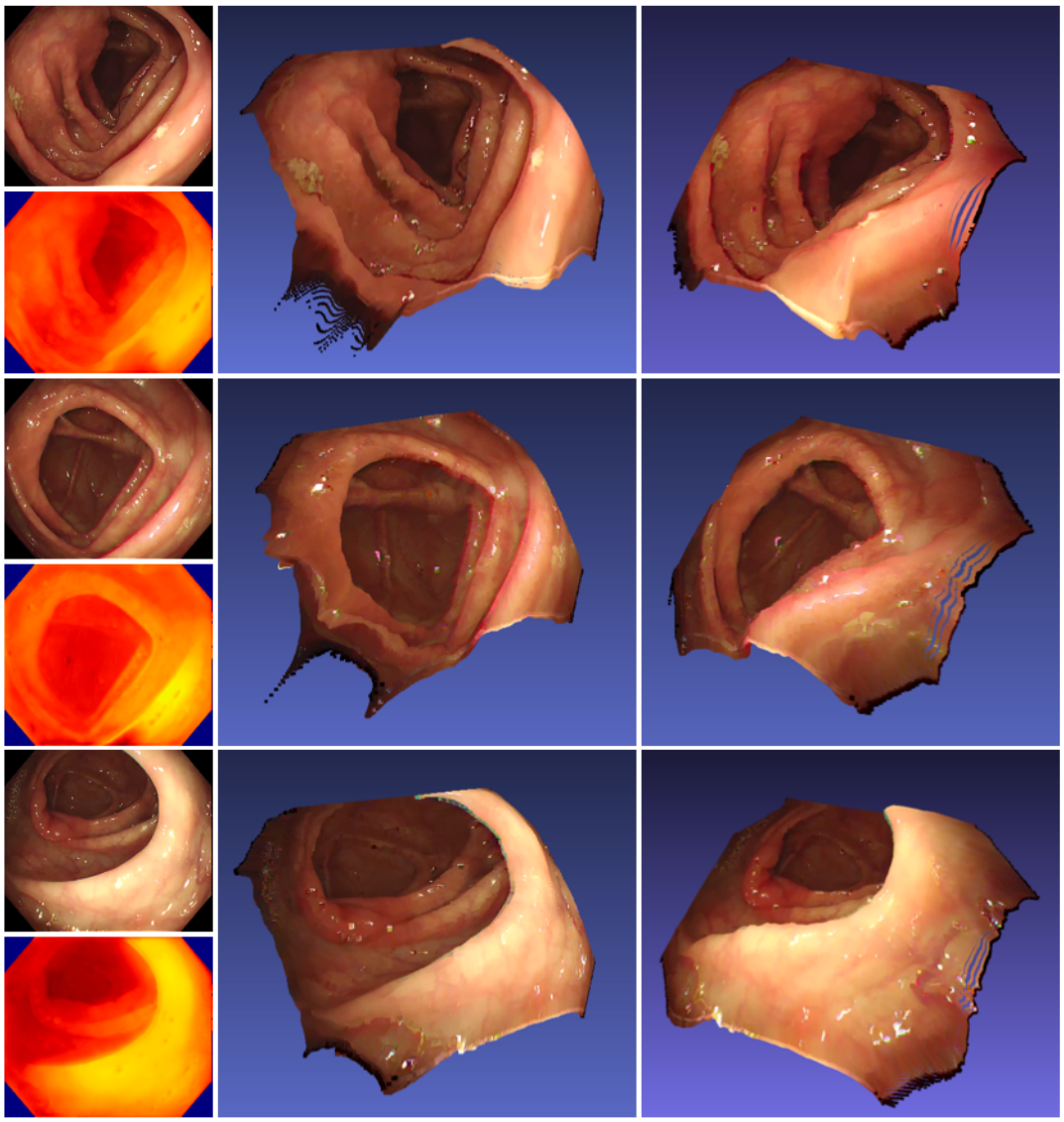


Fig. 5.6 This figure shows the predicted depth and the corresponding 3D reconstruction obtained for test images.

(note that the images with plastic cap were discarded for the same reason). Another challenge can be the light speckles. Since the model uses brightness as one of the major cues for depth, the predicted depths at the speckles are smaller than those of the surrounding surface.

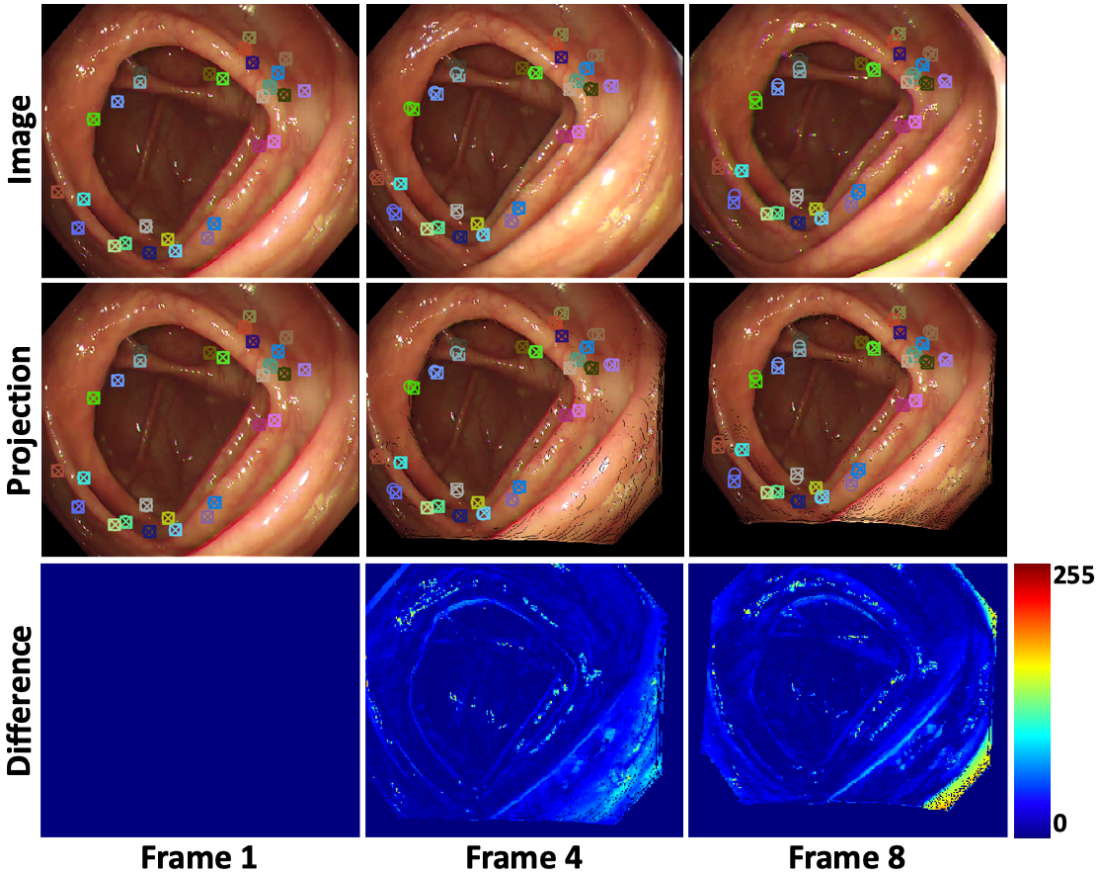


Fig. 5.7 The first row shows the original images. The second row shows the projection of the 3D reconstruction predicted on the first frame. On each image, three sets of features are plotted. Firstly, the features' pixel coordinates identified in the feature detection are plotted with 'X's. Secondly, the projections of the ground truth 3D feature coordinates are plotted with squares. For each square in the first frame, the predicted depth at that pixel can specify its 3D coordinates. These predicted 3D feature coordinates are projected to each image and are plotted with circles. The fact that the circles are close to the squares suggests that the predicted 3D feature coordinates are close to the ground truth. The third row shows the photometric difference between the images and the projections (in Jet colour scheme).

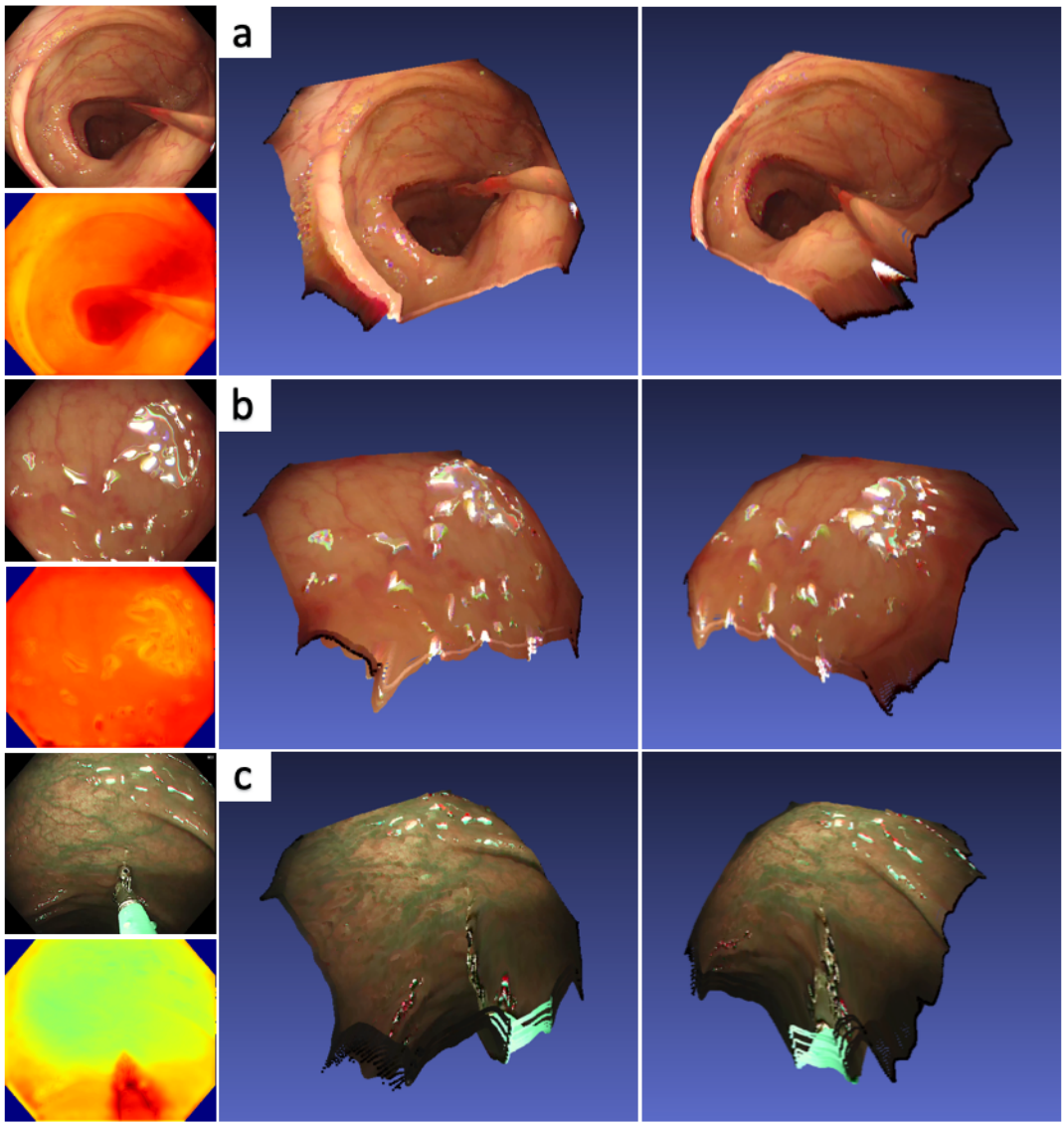


Fig. 5.8 This figure shows examples of challenging images, for which the model prediction was not accurate. Figure (a) contains water jet. The model treats the water jet as part of the colon surface, and makes prediction based on its brightness. Figure (b) shows an image with strong speckles. The depth predicted at the speckles is smaller than those of the surrounding pixels. This suggests that pixel brightness is one of the major depth cues used by the model. Figure (c) shows an image where the polyp removal device is visible. The model predicts that the device is further away than the colon wall.

Chapter 6

Conclusion

The aim of this project was to create a monocular depth estimation framework for colonoscopy images. We generated accurate ground truth relative depth from colonoscopy images and used the data to train a monocular depth estimation model. After finding the best-performing model configuration, we evaluated its accuracy on the test set.

Ground truth was generated by applying structure from motion [36] on short sequences of consecutive colonoscopy images. The step-size between the frames was selected to be short enough to assume rigid surface, but long enough to ensure sufficient camera motion. While it is challenging to obtain dense reconstruction of the whole colon, we showed that it is possible to obtain accurate sparse reconstruction of the local colon surface. When applying structure from motion, we set several constraints (e.g. on the minimum feature track length) to obtain accurate reconstruction, while sacrificing density. By measuring the reprojection error and the camera motion, we showed that the generated ground truth is accurate.

Then, the obtained sparse ground truth is used to train a convolutional encoder-decoder network with single output channel. To account for the sparsity and arbitrary scale of the ground truth, we introduced a scale-invariant training loss that calculates the squared error after scale-matching. We showed that the scale-matching helps the model to learn the relative depth of the scene and to achieve better prediction accuracy. We also showed that the U-Net architecture, due to its unique skip connections, can produce accurate prediction where the structural characteristics of the surface such as the circular folds are clearly visible. Lastly, we showed that the encoder weights pre-trained on a large-scale image dataset lead to poor performance, suggesting that the model should learn the visual features specific to the colonoscopy images.

Lastly, we evaluated the best-performing model (randomly initialised U-Net trained with scale-invariant loss) on the test set. We introduced three metrics - RMSE, REL, and aRMSE - that can measure the prediction accuracy at the pixels with ground truth. The RMSE and REL

on test set are 9.99 and 0.0566, respectively. The true scale RMSE (or aRMSE) estimated on a subset of 20 sequences is 2.7*mm*.

Major contributions of this project include (1) an accurate monocular depth estimation model trained with scale-invariant loss, (2) large-scale dataset of monocular colonoscopy images and the corresponding relative depth, and (3) quantitative and qualitative evaluation of the model. Possible extensions of this work include following:

- **Absolute scale estimation.** One of the limitations of the proposed method is that it can only predict the relative depth of the scene. In order to predict the depth in absolute scale (e.g. in *mm*), the ground truth should be converted into physical unit. This can be achieved by measuring the size of a removed polyp. Then, for the sequence where such polyp was visible, it is possible to convert the relative depth into absolute scale, so that it agrees with the measured polyp size.

- **Approximate partial colon reconstruction.** The 3D reconstructions predicted at the neighbouring frames can be connected (e.g. based on the feature matches). Two point clouds should be warped properly to account for the non-rigid motion. Such process will give an approximate partial reconstruction of the colon.

References

[1] Bae, S. Y., Korniski, R. J., Shearn, M., Manohara, H. M., and Shahinian, H. (2016). 4-mm-diameter three-dimensional imaging endoscope with steerable camera for minimally invasive surgery (3-d-marvel). *Neurophotonics*, 4(1):011008.

[2] Barnes, C., Shechtman, E., Finkelstein, A., and Goldman, D. B. (2009). PatchMatch: A randomized correspondence algorithm for structural image editing. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 28(3).

[3] Cancer Research UK (2016). Bowel cancer statistics. https://www.cancerresearchuk.org/health-professional/cancer-statistics/statistics-by-cancer-type/bowel-cancer/, accessed 23 July 2019.

[4] Chaurasia, A. and Culurciello, E. (2017). Linknet: Exploiting encoder representations for efficient semantic segmentation. In *2017 IEEE Visual Communications and Image Processing (VCIP)*, pages 1–4. IEEE.

[5] Chen, G., Pham, M., and Redarce, T. (2009). Sensor-based guidance control of a continuum robot for a semi-autonomous colonoscopy. *Robotics and Autonomous Systems*, 57(6):712 – 722.

[6] Chollet, F. et al. (2015). Keras. https://keras.io.

[7] Consolo, P., Luigiano, C., Strangio, G., Scaffidi, M. G., Giacobbe, G., Di Giuseppe, G., Zirilli, A., and Familiari, L. (2008). Efficacy, risk factors and complications of endoscopic polypectomy: ten year experience at a single center. *World journal of gastroenterology: WJG*, 14(15):2364.

[8] FFmpeg Developers (2017). FFmpeg. https://ffmpeg.org.

[9] Hartley, R. I. (1997). In defense of the eight-point algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(6):580–593.

[10] Hartley, R. I. and Sturm, P. (1997). Triangulation. *Computer Vision and Image Understanding*, 68(2):146–157.

[11] Hartley, R. I. and Zisserman, A. (2004). *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 052154051, second edition.

[12] Hazzat, S. E., Merras, M., Akkad, N. E., Saaidi, A., and K., S. (2018). 3d reconstruction system based on incremental structure from motion using a camera with varying parameters. *The Visual Computer*, 34(10):1443–1460.

[13] He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[14] Hong, D., Tavanapong, W., Wong, J., Oh, J., and De Groen, P. C. (2014). 3d reconstruction of virtual colon structures from colonoscopy images. *Computerized Medical Imaging and Graphics*, 38(1):22–33.

[15] Hou, Y., Dupont, E., Redarce, T., and Lamarque, F. (2014). A compact active stereovision system with dynamic reconfiguration for endoscopy or colonoscopy applications. In Golland, P., Hata, N., Barillot, C., Hornegger, J., and Howe, R., editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2014*, pages 448–455, Cham. Springer International Publishing.

[16] International Agency for Research on Cancer (2018). Globocan. http://gco.iarc.fr/today/data/factsheets/cancers/39-All-cancers-fact-sheet.pdf, accessed 23 July 2019.

[17] Itoh, H., Roth, H. R., Lu, L., Oda, M., Misawa, M., Mori, Y., Kudo, S.-e., and Mori, K. (2018). Towards automated colonoscopy diagnosis: binary polyp size estimation via unsupervised depth learning. In *International conference on medical image computing and computer-assisted intervention*, pages 611–619. Springer.

[18] Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

[19] Li, H. and Hartley, R. (2006). Five-point motion estimation made easy. In *18th International Conference on Pattern Recognition (ICPR'06)*, volume 1, pages 630–633.

[20] Lin, T.-Y., Dollár, P., Girshick, R., He, K., Hariharan, B., and Belongie, S. (2017). Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125.

[21] Lohsiriwat, V. (2010). Colonoscopic perforation: incidence, risk factors, management and outcome. *World journal of gastroenterology*, 16(4):425–430.

[22] Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110.

[23] Mahmood, F. and Durr, N. J. (2018). Deep learning and conditional random fields-based depth estimation and topographical reconstruction from conventional endoscopy. *Medical image analysis*, 48:230–243.

[24] Mapillary (2017). Opensfm. https://github.com/mapillary/OpenSfM.

[25] McConnell, R. K. (1986). Method of and apparatus for pattern recognition. *United States Patent*, US4567610A.

[26] Mikolajczyk, K. and Schmid, C. (2002). An affine invariant interest point detector. *European Conference on Computer Vision*, pages 128–142.

[27] Mir, F. A., Boumitri, C., Ashraf, I., Matteson-Kome, M. L., Nguyen, D. L., Puli, S. R., and Bechtold, M. L. (2017). Cap-assisted colonoscopy versus standard colonoscopy: is the cap beneficial? a meta-analysis of randomized controlled trials. *Annals of gastroenterology*, 30(6):640–648.

[28] Muja, M. and Lowe, D. G. (2014). Scalable nearest neighbor algorithms for high dimensional data. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 36.

[29] Oquab, M., Bottou, L., Laptev, I., and Sivic, J. (2014). Learning and transferring mid-level image representations using convolutional neural networks. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1717–1724.

[30] Parot, V., Lim, D., González, G., Traverso, G., Nishioka, N. S., Vakoc, B. J., and Durr, N. J. (2013). Photometric stereo endoscopy. *Journal of biomedical optics*, 18(7):076017.

[31] Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P. (2007). *Numerical Recipes 3rd Edition: The Art of Scientific Computing*. Cambridge University Press, New York, NY, USA, 3 edition.

[32] Ronneberger, O., Fischer, P., and Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer.

[33] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei-Fei, L. (2015). Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252.

[34] Schmalz, C., Forster, F., Schick, A., and Angelopoulou, E. (2012). An endoscopic 3d scanner based on structured light. *Medical image analysis*, 16(5):1063–1072.

[35] Triggs, B., McLauchlan, P. F., Hartley, R. I., and Fitzgibbon, A. W. (2000). Bundle adjustment — a modern synthesis. In Triggs, B., Zisserman, A., and Szeliski, R., editors, *Vision Algorithms: Theory and Practice*, pages 298–372, Berlin, Heidelberg. Springer Berlin Heidelberg.

[36] Ullman, S. (1979). The interpretation of structure from motion. *Proceedings of the Royal Society of London*, 203(1153):405–426.

[37] Yakubovskiy, P. (2019). Segmentation models. https://github.com/qubvel/segmentation_models.

[38] You and Colonoscopy (2016). What happens during and after a colonoscopy? https://youtu.be/mh90RPA-C10, accessed 6 August 2019.

[39] 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*, pages 1851–1858.