UNIVERSITY OF MISKOLC



Faculty of Mechanical Engineering and Informatics

Efficient Visual Data Encoding Methods for Robust Spatial Perception Robotic Systems

PH.D. DISSERTATION

Prepared by:

Omar Mohammed Salih Alsabaawi
M.Sc. in Computer Technology Engineering

József Hatvany Doctoral School for Computer Science and Engineering

Research area: Computer Supported Production Systems
Research group: Automatization and Control Systems

Head of doctoral school: **Prof. Dr. László Kovács**
Supervisor: **Prof. Dr. József Vásárhelyi**

Miskolc 2026

Statement of Originality

The research presented in this dissertation is the original work of Omar Mohammed Salih Alsabaawi, except where explicitly indicated. In cases where derivations or results are based wholly or in part on the contributions of other sources, proper acknowledgment and full referencing to the original authors have been provided. This dissertation has not been submitted previously, in whole or in part, for the award of any degree, nor is it currently under consideration for such by any other institution.

Signed: Omar Mohammed Salih Alsabaawi

Declaration of Artificial Intelligence Assistance

The author declares that Overleaf was used as the LaTeX writing and typesetting environment during the preparation of this dissertation. The integrated language support tool was used solely for language enhancement, grammar checking, spelling correction, and improving the clarity of the writing.

No generative artificial intelligence tools were used to generate or manipulate research data, perform analysis, or produce original academic content. The intellectual contribution, arguments, and conclusions presented in this dissertation remain the sole work of the author.

Signed: Omar Mohammed Salih Alsabaawi

Supervisor's endorsement

During the course of the research, the candidate has successfully achieved the objectives that were set to develop efficient and robust encoding methods to improve 2D image compression used in spatial perception robotic systems. This was achieved by adapting and proposing novel techniques through the following points:

- The candidate developed DFT- and DCT-based visual data encoding methods for 3D reconstruction, incorporating coefficient reduction, optimized quantization, coefficient elimination, and lossless entropy coding to achieve high compression ratios while preserving the information required for accurate reconstruction from structured-light and multi-view images.
- The candidate developed multi-level visual data encoding methods integrated into monocular ORB-SLAM to improve compression efficiency while preserving localization, mapping, and trajectory estimation accuracy. These methods were further incorporated into an edge-assisted multi-robot VSLAM architecture to reduce communication and memory requirements while maintaining localization accuracy and global map consistency.
- The proposed visual data compression framework was implemented using a hardware-software co-design on the AMD Kria KV260 FPGA platform. Dedicated compression and decompression accelerators were experimentally evaluated in terms of processing performance, resource utilization, scalability, and power consumption, demonstrating the feasibility of the framework for resource-constrained robotic spatial perception systems.

The applicant has proved that he is a dedicated researcher.

Signed: Prof. Dr. József Vásárhelyi

Acknowledgements

First and foremost, I would like to express my profound gratitude to my supervisor, Prof. Dr. József Vásárhelyi, at the Institute of Automation and Info-Communication, University of Miskolc, for his exceptional guidance, unwavering support, and invaluable mentorship throughout my PhD journey. His continuous encouragement, patience, scientific insight, and constructive feedback have been essential to the completion of this dissertation. Beyond scientific supervision, I deeply appreciate his kindness, personal support, sincere advice, and the understanding and care he has shown throughout my doctoral studies. His trust, dedication and unwavering belief in my abilities will always be deeply appreciated and respected.

In addition, I would like to express my sincere gratitude to Dr. Attila Trohák, Head of the Institute of Automation and Info-Communication, as well as to my colleagues at the Institute, for their support, understanding, and encouragement throughout my academic journey. I would also like to extend my appreciation to Prof. Dr. László Kovács, Head of the József Hatvany Doctoral School of Computer Science and Engineering, for his support and guidance.

I am also sincerely grateful to the Stipendium Hungaricum Programme, the University of Miskolc, the Ministry of Higher Education and Scientific Research of Iraq, and Northern Technical University for providing me with the valuable opportunity, financial support and continuous encouragement to pursue my doctoral studies in Hungary. Their support has greatly contributed to the successful completion of my doctorate studies.

Finally, and above all, all praise and thanks are due to Allah for granting me strength, patience, guidance, and perseverance throughout this journey. Without His blessings and mercy, the completion of this work would not have been possible.

Dedication

To my beloved parents, whose sacrifices, prayers, and guidance have shaped my path.

To my brothers and sisters, for their constant encouragement and assistance.

To my lovely wife, for her patience, unwavering support, and endless love.

To my dear son, Adam, who brings joy and meaning to every moment.

I dedicate this work with love, gratitude, and respect.

Abstract

The rapid development of edge-assisted robotic and autonomous perception systems has increased the demand for efficient visual data processing with limited computational, memory, energy, and communication resources. This dissertation investigates efficient visual data encoding for resource-constrained robotic perception systems, with particular emphasis on monocular Visual Simultaneous Localization and Mapping (VSLAM), collaborative multi-robot navigation, and Three-Dimensional (3D) reconstruction applications. Unlike conventional image compression approaches that focus mainly on visual quality metrics, this research considers the impact of compression on robotic perception tasks, including feature preservation, localization accuracy, mapping consistency, communication efficiency, and reconstruction quality. The dissertation proposes a unified transform-domain visual encoding framework integrated directly into robotic perception pipelines. A novel multi-level encoding strategy is developed for monocular Oriented FAST and Rotated BRIEF Simultaneous Localization and Mapping (ORB-SLAM) systems to reduce communication overhead while preserving essential feature information required for reliable localization and mapping. The proposed framework includes optimized transform-domain processing, coefficient reduction, quantization, and compact encoding mechanisms designed for bandwidth-constrained environments. In addition, an edge-assisted architecture for collaborative multi-robot VSLAM is introduced, where computational tasks are efficiently partitioned between mobile robots and edge devices to reduce onboard processing requirements and improve system scalability. The work is further extended toward structured-light and multi-view image compression for 3D reconstruction using optimized Discrete Cosine Transform (DCT)- and Discrete Fourier Transform (DFT)-based approaches combined with frequency-domain coefficient minimization techniques. The proposed methods were implemented and evaluated

using MATLAB-based simulation environments, ORB-SLAM frameworks, and Field-Programmable Gate Array (FPGA)-based hardware-software co-design platforms for embedded edge systems. The experimental evaluation was conducted under different compression and quantization settings using metrics including compression ratio, Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index Measure (SSIM), Root Mean Square Error (RMSE), accuracy of trajectory estimation, memory usage, and computational complexity. Comparative analysis against JPEG and JPEG2000 demonstrates that the proposed methods achieve higher compression efficiency while preserving Oriented FAST and Rotated BRIEF (ORB) feature quality and trajectory consistency more effectively under high compression conditions. For 3D reconstruction applications, the proposed approaches achieved high compression ratios while maintaining accurate geometric reconstruction and preserving important structural information. Furthermore, the dissertation presents an FPGA-based implementation of the proposed visual encoding pipeline using an efficient programmable logic and processing system partitioning strategy suitable for embedded robotic systems. The hardware implementation demonstrates balanced resource utilization, reduced communication overhead, and real-time processing capability, confirming the practicality of the proposed framework for resource-constrained robotic perception platforms. The findings of this dissertation contribute to the development of scalable, bandwidth-aware, and edge-assisted robotic vision systems capable of operating efficiently in real-world environments with limited computational and communication resources.

Table of Contents

Supervisor's endorsement ii

Acknowledgements iii

Dedication iv

Abstract v

List of Figures x

List of Tables xiv

Abbreviations xvi

1 Introduction 1

1.1 Background and Motivation 1
1.2 Problem Statement 5
1.3 Research Objectives 7
1.4 Dissertation Outline 9

2 Baseline DFT-Domain Compression Method for Spatial Perception 11

2.1 Literature Background and Motivation 11
2.2 The Proposed Compression Algorithm 14
2.2.1 Discrete Fourier Transform and Quantization 14
2.2.2 Coefficient Reduction Algorithm 16
2.2.3 Encoding of Zero and Nonzero values 18
2.3 The Decompression Algorithm 19
2.4 Experimental results 21
2.4.1 Results for grayscale and color images 21
2.4.2 Results for Structured-light Images and 3D Surfaces 22
2.5 Discussion and Comparative Analysis 24
2.6 Thesis I. 27

2.7 Chapter Findings 27

3 Beyond JPEG Limitations: Improved Image Compression for 3D Reconstruction 29

3.1 Literature Background and Introduction 29
3.2 The Proposed Compression Pipeline 32
3.2.1 Discrete Cosine Transform 32
3.2.2 Optimized Quantization 33
3.2.3 First Order Difference Processing 34
3.3 The Proposed Decompression Pipeline 35
3.4 Experimental results 36
3.4.1 Comparison with the JPEG standard for structured-light images 37
3.4.2 Comparison with JPEG and 3D Object (.OBJ) File Sizes . . . 37
3.5 Thesis II. 38
3.6 Chapter Findings 39

4 End-to-End Multi-Level Encoding for Robust Monocular Visual ORB-SLAM 41

4.1 Introduction 41
4.2 Literature Background: Visual Simultaneous Localization and Mapping (Visual SLAM) 42
4.2.1 Monocular VSLAM 45
4.3 The Proposed End-to-End Frame Encoding Pipeline 48
4.3.1 Multi-level Encoding Module on the Mobile Robot 48
4.3.2 The Decoding Module on the Edge Device 51
4.4 Experimental Results and Discussion 52
4.5 Thesis III. 60
4.5.1 Sub-thesis III.1 60
4.5.2 Sub-thesis III.2 60
4.6 Chapter Findings 60

5 Visual Data Encoding in Multi-Robot Visual SLAM Systems 63

5.1 Literature Background and Introduction 63
5.2 Methodology 66
5.2.1 Proposed Multi-Robot V-SLAM System Architecture 66
5.2.2 Proposed Communication Module 67
5.3 Results and Discussion 71
5.4 Thesis IV. 80

5.5 Chapter Findings 81

6 Efficient FPGA Implementation of a Visual Data Encoding/Decoding Pipeline 83

6.1 Introduction and Literature Background 83
6.2 System Design and Implementation 85
6.2.1 Hardware-Software Co-Design Strategy 85
6.3 Encoding and Decoding Pipeline 86
6.4 Implementation results and Discussion 89
6.5 Thesis V. 94
6.6 Chapter Findings 94

7 Summary and Future Research Directions 95

7.1 Dissertation Summary 95
7.2 Future Research Directions 96
7.3 Theses 98
7.3.1 Thesis I. 98
7.3.2 Thesis II. 98
7.3.3 Thesis III. 99
7.3.4 Thesis IV. 100
7.3.5 Thesis V. 100

Author's publications 101

References 103

Appendix A Chapter 2 Supplementary Results 115

Appendix B Chapter 3 Supplementary Results 127

Appendix C Chapter 4 Supplementary Results 143

Appendix D Chapter 5 Supplementary Results 153

Appendix E Chapter 6 Supplementary Material 161

List of Figures

2.1 The proposed compression method. 14
2.2 Application of the DFT to a 4×4 image block. 16
2.3 Quantization and rounding of real and imaginary components 16
2.4 Decomposition of the 4×4 DFT coefficient matrix into real and imaginary matrices. 16
2.5 Illustration of the matrix reduction method for an $M \times N$ matrix. . . 17
2.6 Separating zero and nonzero values from each HFC matrix into a zero matrix and a value matrix. 19
2.7 Sequential search mechanism used during decompression. 20
2.8 The proposed decompression steps. 20
2.9 (a) The 3D Scanner developed by the GMPR group, (b) a 2D image captured by the scanner, and (c) the 3D surface patch reconstructed from the 2D image. 22
2.10 Original 2D images with different sizes used by the proposed compression method. 23
2.11 Decompressed greyscale images using JPEG, the quality of the decompressed images varies compared with the proposed approach according to RMSE and PSNR. 25
2.12 Decompressed color images by JPEG, the decompressed image (Boeing) has lower quality compared with the proposed approach according to RMSE and PSNR. However, the approach could not reach the JPEG compression level for Baghdad's image on the right. 25

3.1 The proposed DCT-based compression pipeline. 32
3.2 The coefficient array format. 35
3.3 The Proposed compression algorithm. 35

4.1 Edge assisted VSLAM paradigms: (a) Analyze then Compress (ATC), and (b) Compress then Analyze (CTA). 44
4.2 Illustration of common visual features, including corners, edges, and textured blocks, used in feature extraction techniques. 46

4.3 ORB-SLAM Architecture. 48
4.4 The proposed image compression pipeline. 50
4.5 Single-level encoding implementation (the symbols (K_i) denote the encoding keys used during the coefficient reduction process). 50
4.6 Two-level encoding implementation. The same encoding key set (K_i) is reused at both encoding levels in the proposed implementation. . . 51
4.7 The proposed image decoding pipeline. 52
4.8 Mechanism of coefficient estimating using sequential search. 53
4.9 Standard ORB-SLAM trajectory estimation performance (RMSE = 0.22114). 54
4.10 Computational complexity of the three implemented systems. 59

5.1 Proposed multi-robot ORB-SLAM architecture. 67
5.2 Proposed communication pipeline. 68
5.3 Proposed sequential search mechanism for coefficient decoding. 69
5.4 Comparison of the computational complexity of all implemented systems. 80

6.1 Decomposition of the 2D DCT into horizontal and vertical 1D DCT stages. 87
6.2 Graph of RMSE for VSLAM trajectory vs. compression ratio of the three implemented encoding systems. 92
6.3 Resource utilization of the implemented FPGA-based system. 93
6.4 Power consumption of the implemented FPGA-based system. 93

A.1 (a) and (b) Compression results for the Lion and Apple images using the proposed method with different quantization factors. 116
A.2 (a) and (b): Boeing 777 and Baghdad color images are compressed by the proposed method using different quantization factors. 117
A.3 (a) and (b) Decompressed 2D Corner image used for 3D reconstruction. The 3D mesh surface, consisting vertices and triangles, is successfully reconstructed without significant distortion at high compression ratios of up to 98.6%. 118
A.4 (a) and (b) Decompressed 2D Face-1 image used for 3D reconstruction. The 3D mesh surface, consisting vertices and triangles, is successfully reconstructed without significant distortion at high compression ratios of 98.6%. 119

A.5 (a) and (b) Decompressed 2D Face-2 2D image used for 3D reconstruction. 3D mesh surface. The 3D mesh surface, consisting vertices and triangles, is successfully reconstructed without significant distortion at high compression ratios of 98.5%. 120
A.6 3D reconstruction from JPEG compressed images. In (a), reconstruction was possible but but exhibited significant artifacts. In (b), 3D reconstruction was not possible because the images were too degraded. 125

B.1 (a, b, c, and d) Decompressed "Face1" image converted successfully to a 3D surface at higher compression ratio (99%). 128
B.2 (a, b, c, and d) Decompressed "Face2" image converted successfully to a 3D surface at higher compression ratio (over 98%). 129
B.3 (a, b, c, and d) Decompressed "Face3" image converted successfully to a 3D surface at higher compression ratio (over 98%). 130
B.4 Decompressed 48 frames of Apple images, dimensions for each image $= 1936 \times 1296$. 131
B.5 (a) and (b) show the 3D object and mesh surface for the decompressed Apple images by the proposed compression method at higher compression ratio (97.2%). 132
B.6 Decompressed 42 frames of Soft Toy images, dimensions for each image $= 1936 \times 1296$. 133
B.7 (a) and (b) show the 3D object and the mesh surface for the compressed Soft Toy images by the proposed method at higher compression ratio (97.6%). 134
B.8 (a) and (b) 3D mesh surface reconstruction for "Face1" comparison of traditional JPEG standard versus proposed method. 135
B.9 (a) and (b) 3D mesh surface reconstruction for "Face2" comparison of traditional JPEG standard versus proposed method. 136
B.10 (a) and (b) 3D mesh surface reconstruction for "Face3" comparison of traditional JPEG standard versus proposed method. 137
B.11 (top and middle) 3D object and 3D mesh for "Apple" image decompressed by the traditional JPEG algorithm at higher compression ratio of 94%, (bottom) 3D mesh reconstruction at higher compression ratio of 97.2% by the proposed method. 139
B.12 (top and middle) 3D object and 3D mesh for "Soft Toy" image decompressed by the traditional JPEG algorithm at higher compression ratio of 97%, (bottom) 3D mesh reconstruction at higher compression ratio of 97.6% by the proposed method. 140

B.13 (a and c) Perceptual assessment: the decompressed image quality shows artifacts at higher compression ratios by the JPEG algorithm, while the other decompressed images (b and d) kept the quality at higher compression ratios by the proposed algorithm. 142

C.1 ORB-SLAM trajectory estimation performance adopting JPEG compression. 144
C.2 Trajectory estimation performance of ORB-SLAM system adopting JPEG 2000. 145
C.3 Multi-level based-ORB-SLAM trajectory estimation performance at lower QFs and CRs. 146
C.4 Multi-level based-ORB-SLAM trajectory estimation performance at higher QFs and CRs. 147

D.1 Estimated trajectory of Robot-1 compared to ground truth. 154
D.2 Estimated trajectory of Robot-2 compared to ground truth. 155
D.3 Proposed multi-robot ORB-SLAM trajectory estimation using uncompressed visual data and the resulting global fused map generated from the combined robot trajectories. 156
D.4 Visualization of global map trajectories at high quantization (Q=100). 157
D.5 Visualization of global map trajectories at severe quantization (Q=120).158

E.1 Hardware–software co-design architecture of the proposed FPGA-based visual data encoding and decoding system. 162
E.2 Implementation of the proposed FPGA-based system using AMD Vivado.163

List of Tables

2.1 Compression results for grayscale images. 21
2.2 Compression results for color images. 21
2.3 Compressed 2D structured-light images using the proposed method. . 23

3.1 Compression performance of the proposed method for structured-light images. 36
3.2 2D multi-view images Compressed using the proposed method at high compression ratios. 37

4.1 Encoding performance metrics of JPEG integrated into ORB-SLAM. 56
4.2 Encoding performance metrics of JPEG2000 integrated into ORB-SLAM. 56

5.1 Communication performance key metrics for the proposed multi-robot ORB-SLAM architecture. 73
5.2 Communication performance key metrics for a corresponding JPEG-based architecture. 75
5.3 Communication performance key metrics for a corresponding JPEG2000-based multi-robot architecture. 76

6.1 Encoding performance of the JPEG system (CPU-based implementation) 90
6.2 Encoding performance of the JPEG2000 system (CPU-based floating-point implementation) 90
6.3 Encoding performance of the proposed method (FPGA-based fixed-point implementation) 91

A.1 Comparison between the DFT and the proposed method (DFT with the matrix reduction algorithm) in terms of image quality and compressed size. 121
A.2 Comparison between JPEG and the proposed method in terms of image quality and compressed size. 122

A.3 Comparison between the conventional matrix reduction approach [39] and the proposed approach in terms of execution time. 123
A.4 Comparison between the method in [38] and the proposed approach in terms of image quality and compressed size. 124

B.1 Results of the conventional JPEG standard and OBJ file sizes for gray-scale structured-light images compared with the proposed algorithm. 138
B.2 Results of the conventional JPEG standard and OBJ file sizes for multiple-viewpoint images compared with the proposed algorithm. . . 141

C.1 Encoding performance metrics of the ORB-SLAM integrating the proposed multi-level method. 148
C.2 Comparison of compression performance metrics between JPEG, JPEG2000, and the proposed methods at approximately equal compression ratios. 149
C.3 Complexity analysis metrics of JPEG encoding (processing frame sequence 1132). 150
C.4 Complexity analysis metrics of JPEG2000 (processing frame sequence 1132). 151
C.5 Complexity analysis metrics of the proposed method using a single-level encoding (processing frame sequence 1132). 152

D.1 Compression performance of the proposed methods, JPEG, and JPEG2000 for frame 2068 (900 KB). 159
D.2 Complexity analysis metrics of the proposed methods for processing a single frame (frame sequence 2068) 160

Abbreviations

2D:	Two-Dimensional
3D:	Three-Dimensional
AC:	Alternating Current (coefficients)
ATC:	Analyze Then Compress
AV:	Autonomous Vehicle
AXI4:	Advanced eXtensible Interface version 4
BoW:	Bag of Words
bpp:	its per pixels
BRAM:	Block Random Access Memory
BRIEF:	Binary Robust Independent Elementary Features
CAD:	Computer-Aided Design
CNN:	Convolutional Neural Networks
CORB-SLAM:	Collaborative Oriented FAST and Rotated BRIEF-SLAM
CPLD:	Complex Programmable Logic Device
CPU:	Central Processing Unit
CR:	Compression Ratio
CTA:	Compress Then Analyze
DC:	Direct Current (coefficient)
DCT:	Discrete Cosine Transform
DFT:	Discrete Fourier Transform
DICT:	Dictionary Table
DL:	Deep Learning
DMA:	Direct Memory Access
DNN:	Deep Neural Networks
DPCM:	Differential Pulse Code Modulation
DSP:	Digital Signal Processing
DST:	Discrete Sine Transform
DWT:	Discrete Wavelet Transform
ESC:	Escape mechanism
FAST:	Features from Accelerated Segment Test

FFT:	Fast Fourier Transform
FMS:	Fast Match Search
FOSS:	Free and Open Source Software
FPGA:	Field Programmable Gate Array
GMPR:	Group of Modeling and Pattern Recognition
GPU:	Graphical Processing Unit
HFC:	High Frequency Coefficients
HEVC:	High Efficiency Video Coding
HLS:	High-Level Synthesis
HVS:	Human Visual System
IDCT:	Inverse Discrete Cosine Transform
IDFT:	Inverse Discrete Fourier Transform
IDWT:	Inverse Discrete Wavelet Transform
IDE:	Integrated Development Environment
IoT:	Internet of Things
IP:	Intellectual Property
JPEG:	Joint Photographic Experts Group
JPEG-LS:	JPEG Lossless
LFC:	Low Frequency Coefficients
LiDAR:	Light Detection and Ranging
LUT:	Lookup Table
MATLAB:	Matrix Laboratory
MMS:	Minimize-Matrix Size
NIC:	Neural Image Compression
OBJ:	Wavefront Object (3D file format)
ORB:	Oriented FAST and Rotated BRIEF
ORB-SLAM:	Oriented FAST and Rotated BRIEF-Simultaneous Localization and Mapping
ORB-SLAM2:	Oriented FAST and Rotated BRIEF-SLAM version 2
ORB-SLAM3:	Oriented FAST and Rotated BRIEF-SLAM version 3
PL:	Programmable Logic
PNG:	Portable Network Graphics
PS:	Processing System
PSNR:	Peak Signal-to-Noise Ratio
QF:	Quantization Factor
RAM:	Random Access Memory
RGB:	Red, Green, and Blue
RGB-D:	Red, Green, Blue and Depth

RLE:	Run-Length Encoding
RMSE:	Root Mean Square Error
ROS2:	Robot Operating System 2
SIFT:	Scale-Invariant Feature Transform
SLAM:	Simultaneous Localization and Mapping
SSIM:	Structural Similarity Index Measure
SURF:	Speeded Up Robust Features
TUM:	Technical University of Munich
UAV:	Unmanned Autonomous Vehicle
UGV:	Unmanned Ground Vehicle
VCM:	Video Coding for Machines
ViT:	Vision Transformer
VLSI:	Very Large Scale Integration
VSLAM:	Visual Simultaneous Localization and Mapping
VVC:	Versatile Video Coding

Chapter 1

Introduction

1.1 Background and Motivation

Computer vision has become a key enabling technology for intelligent robotic and autonomous systems because it allows machines to extract operationally meaningful information from visual scenes rather than merely recording images. Vision systems are now used in domestic and assistive robotic systems, autonomous vehicles, automated inspection, biometric systems, and other machine perception applications where cameras and algorithms jointly support recognition, localization, monitoring, and decision-making [1]. In autonomous driving, for example, detection of road objects is a necessary prerequisite for higher-level functions such as tracking, free space identification, risk assessment, and path planning [2]. In mobile robotics, visual and spatial mapping systems support environmental exploration, localization, search and rescue, motion planning, monitoring, and operator awareness. These examples show that visual data are no longer only media for human viewing, but they are increasingly the primary input for autonomous interpretation, spatial reasoning, and robotic actions. Three-dimensional maps further improve the ability of robots and operators to understand real environments, but they also introduce heavy data and processing requirements, especially when real-time performance is expected on mobile or low-cost platforms [3]. Similar requirements appear in 3D reconstruction, where the objective is not only to preserve the appearance of images but also to recover geometry, depth, structure, and surface continuity from visual observations [4, 5, 6]. Thus, for robotic perception, the value of an image is determined not only by how it

looks to the human observer, but by how effectively it supports feature extraction, correspondence estimation, depth inference, pose estimation, and reconstruction.

The growing dependence on visual sensing has created a serious resource problem. Modern robotics, industrial, biomedical, and cloud-assisted vision systems generate large volumes of visual data that must be stored, transmitted, and processed under practical constraints. Studies on Internet of Things (IoT) and smart camera systems emphasize that distributed sensors often offload data to remote servers,making compression necessary to reduce bandwidth and power consumption before transmission [7]. Deep Learning (DL)-oriented IoT systems face similar pressures because large volumes of visual data must be transmitted or stored before training or inference, while terminal devices often have limited communication bandwidth, energy, storage, and hardware resources [8]. In medical and biomedical analysis, the same dilemma appears on an even larger scale, medical imaging produces substantial data volumes, and cloud-based analysis may require the transmission of large 3D images before 3D object reconstruction can begin to compute [9, 10, 11]. Traditional image compression methods have generally been developed around human visual perception. Their design objective is to reduce the number of bits while maintaining an image that appears visually acceptable to a human observer. This principle is reflected in conventional codecs such as JPEG and JPEG2000, which remove or quantize information according to the assumption about perceptual redundancy and human sensitivity to visual distortion [8, 11, 12]. However, the requirements for machine perception are not identical to the requirements for human viewing. A feature that appears visually insignificant may be critical for Convolutional Neural Networks (CNN), a segmentation algorithm, a Simultaneous Localization and Mapping (SLAM) front-end platform, or a 3D reconstruction algorithm. In contrast, visually pleasing reconstruction does not necessarily guaranty the preservation of image structures that are important for machine-vision analysis, where medical images that appear good according to human visual quality may still produce inferior segmentation accuracy because machine analysis depends on different image information [11].

The same argument is made for image compression oriented to Deep Neural Networks (DNNs), where human-visual-based JPEG is reported to be not optimized for DNN

systems, particularly at high compression ratios [8]. In image classification, early standard compression evaluation showed that compressed images can still be correctly classified by CNNs with substantial compression factors, but also that the relationship between compression parameters, image quality metrics, and classification behavior is task dependent [7]. Further work challenged the simple assumption that compression always harms classification, showing that selecting a suitable JPEG-compressed version of an image can improve classification accuracy while reducing the input bit size [12]. Studies on Vision Transformers (ViTs) similarly reported that encrypted JPEG-compressed images can preserve high classification accuracy while greatly reducing the amount of data under suitable conditions [13]. At the same time, robustness studies show that machine vision models do not respond uniformly to all degradations, such as high levels of noise, blur, missing pixels, or severe exposure changes may significantly degrade performance, whereas compression artifacts may have a more limited or condition-dependent effect [1, 14]. These findings do not imply that compression is harmless; rather, they show that compression must be evaluated according to the target machine task, not only according to visual appearance. The same pattern appears in object detection and industrial machine vision. In the context of autonomous vehicles, compression was found to be less harmful to YOLOv3 detection (You Only Look Once Version 3) when an adequate quality level was maintained, whereas downsampling had a stronger negative effect on overall detection accuracy [2]. In steel surface defect classification, compressed images were used to study CNN performance in an industrial inspection scenario where images may need to be transmitted from limited IoT devices to remote servers. The results showed that classification can remain stable when training and testing conditions are in accordance, but can deteriorate when compression parameters differ, while compression-based augmentation improves robustness [15]. Saliency-guided learned compression for object detection further strengthens the task-aware argument by allocating bit rate according to regions important for machine detection and achieving bit rate reduction without sacrificing detection accuracy [16]. Feature-level compression also supports this direction, as in aircraft detection, compressive sensing was used to reduce feature dimensionality and improve detection speed while maintain-

ing favorable detection accuracy [17]. These works indicate that visual encoding for machines should preserve discriminative and task-relevant information rather than uniformly preserve pixel fidelity.

Biomedical image analysis provides additional evidence that task-driven compression can extend the useful operating range of machine vision. In mitochondria detection from electron microscopy images, the shift from perceptual quality to machine-vision task performance is explicitly identified as a new compression paradigm. Training using compressed images extended the compression range while keeping missed detections low [18]. In mammogram classification, moderate compression did not substantially reduce deep-learning classification performance, while very high compression reduced performance and produced saliency maps that no longer corresponded well to clinically relevant regions [9]. In metastatic cancer detection from histopathological whole-slide images, JPEG2000 compression was found to be acceptable up to a certain compression ratio for a CNN model trained on high-quality images, and training on compressed images improved robustness at the corresponding compression levels [10]. For large-scale image segmentation, fuzzy block-transform compression was used to reduce memory and runtime requirements while maintaining comparable segmentation results at moderate compression levels [19]. These studies are important because they show that machine-oriented compression is not limited to one recognition task, but it is relevant to detection, classification, segmentation, medical analysis, and large-scale image processing.

The transition from human-oriented to machine-oriented compression becomes even more critical when the task is geometric rather than purely semantic. The 3D reconstruction depends on the preservation of structural relationships between images, slices, or views. Compression artifacts, quantization noise, and loss of structural fidelity can affect feature detection, matching, depth estimation, point-cloud generation, and surface reconstruction [5]. Studies on compression optimized for 3D reconstruction therefore argue that classical standards such as JPEG2000 do not account for the end task, and that compression jointly optimized with reconstruction can extend the range of compression rates for which 3D reconstruction remains possible [4]. Recent 3D representation learning also frames reconstruction as a problem

of compressing appearance and geometry into a representation that can predict 3D structure [6]. These works directly support the motivation for visual encoding methods that preserve spatial and geometric information, not merely human-perceived image quality.

Accordingly, the motivation of this dissertation is to develop efficient visual data encoding methods that are aware of robotic perception requirements. The aim is not simply to produce visually acceptable decompressed images, but to preserve the task-relevant information needed by spatial perception algorithms operating under constrained resources.

1.2 Problem Statement

Visual sensing represents one of the most important sensing technologies in modern robotic and intelligent perception systems such as autonomous robots, visual simultaneous localization and mapping (VSLAM) platforms, intelligent transportation systems, industrial inspection systems, collaborative robots, and IoT -based smart environments. They continuously acquire large-scale image and video streams in order to perceive surrounding environments, perform autonomous decision-making, generate detailed 2D or 3D maps or reconstruct objects according to the application. The increasing dependence on vision-based sensing has resulted in substantial growth in visual data generation, creating major challenges in storage, transmission, computational efficiency, and real-time processing. In practical robotic systems, these challenges become even more critical because autonomous platforms usually operate under limited communication bandwidth, constrained hardware resources, strict energy consumption requirements, and low-latency operational conditions [20].

Traditional image and video compression standards such as JPEG [21], JPEG2000 [22], HEVC [23], and VVC [24] were originally designed for human visual consumption. These methods mainly focus on minimizing perceptual distortion while achieving high compression ratios. Conventional codecs typically employ transform-based coding techniques to reduce spatial and temporal redundancies within visual signals. Recent studies in machine vision and intelligent analytics have demonstrated that

visual quality optimized for humans does not necessarily preserve the semantic and structural information required for machine analysis tasks. The study [25] showed that learned nonlinear transform coding could significantly improve compression performance compared to traditional transform-based methods, particularly through end-to-end optimization of latent representations and entropy models. Their work introduced hyper-prior entropy modeling, which enabled more accurate estimation of latent feature distributions and improved rate-distortion optimization. Later [26] extended this concept by integrating auto-regressive context models with hierarchical priors, achieving superior compression efficiency through more accurate entropy estimation. Research [27] further improved the performance of learned image compression by introducing Gaussian mixture models and attention-based contextual entropy estimation. These developments established Neural Image Compression (NIC) as a major alternative to traditional transform-based codecs.

Although these methods achieved high perceptual reconstruction quality, most of them were still optimized according to human visual characteristics rather than machine perception requirements. As emphasized in several recent studies, semantic and geometric information that appears visually insignificant to humans may remain critically important for robotic perception algorithms [20, 28, 29, 30, 31]. For example, edges, corner structures, texture consistency, feature correspondences, and spatial gradients are essential for machine tasks such as object detection, localization, visual odometry, feature matching, scene segmentation, and 3D reconstruction.

Conventional perceptual compression can suppress or distort these structures during quantization and transform coding, causing severe degradation in robotic perception performance despite maintaining visually acceptable image quality. Traditional codecs optimized for human perception are increasingly insufficient for intelligent robotic systems because they fail to preserve the essential semantic and geometric structures for automated analysis. Recent developments in image compression, feature compression, scalable coding, pre-processing-enhanced coding, transferable transformer architectures, and Video Coding for Machines (VCM) collectively establish a new generation of intelligent compression frameworks capable of supporting both machine perception and human interpretation. These advances provide an

important foundation for future robotic systems operating under real-time communication, bandwidth, and computational constraints.

However, many of these approaches rely on computationally intensive learned models or are optimized for specific perception tasks rather than resource-constrained robotic platforms. Consequently, this dissertation focuses on efficient transform-domain visual data encoding methods that emphasize deterministic execution, low computational complexity, FPGA-oriented implementation, and preservation of robotic spatial perception performance. From this perspective, the proposed methods complement recent developments in learned image compression by addressing the practical requirements of real-time edge-assisted robotic systems.

1.3 Research Objectives

The primary objective of this dissertation is to investigate and develop efficient visual encoding and decoding methods for resource-constrained visual robotic perception systems. The research focuses on reducing the computational, memory, storage, and communication requirements associated with visual perception applications while preserving the information necessary for reliable machine perception tasks such as localization, mapping, and three-dimensional reconstruction.

To achieve this objective, the dissertation defines the following research goals:

- Develop efficient transform-domain visual data compression methods capable of achieving high compression ratios while preserving the structured and feature-level information required for spatial perception and machine vision applications.
- Explore alternative transform-domain compression approaches, including Discrete Cosine Transform (DCT) and Discrete Fourier Transform (DFT) based representations, and assess their capability for preserving spatial information required by perception systems.
- Extend the proposed visual encoding concepts to three-dimensional reconstruction applications, including structured-light images and multi-view imaging

systems, and to evaluate the relationship between the compression efficiency and the reconstruction quality.

- Investigate the impact of visual data compression on robotic perception, including trajectory estimation accuracy, mapping consistency, feature preservation, and hardware resource and memory utilization, rather than relying on conventional image quality metrics.
- Investigate the integration of visual data encoding directly within monocular ORB-SLAM systems, and design encoding strategies that reduce storage and communication overhead without significantly affecting feature extraction and matching, pose estimation, localization accuracy, and map quality.
- Design and evaluate a multi-level visual data encoding framework that exploits transform-domain processing, coefficient reduction, quantization, and compact representation techniques to improve the encoding efficiency while maintaining robust VSLAM performance.
- Develop an edge-assisted collaborative multi-robot VSLAM architecture in which computational tasks are efficiently partitioned between mobile robots and edge devices, enabling scalable map fusion, loop closure detection, and global trajectory optimization, under limited communication bandwidth.
- Implement and validate the proposed visual encoding and decoding framework on an embedded hardware platform using FPGA acceleration to demonstrate the feasibility for deployment in resource-constrained robotic systems.
- Compare the proposed methods with conventional image compression standards, particularly the Joint Photographic Experts Group (JPEG) and its successor, JPEG2000, using both image quality metrics and robotic perception performance indicators, thereby establishing their effectiveness for machine-vision-oriented applications.

1.4 Dissertation Outline

The rest of this dissertation is structured as follows: Chapter 2 presents a baseline transform-domain visual data compression method based on the discrete Fourier transform and evaluates its suitability for spatial perception applications. Chapter 3 extends the proposed compression concepts to three-dimensional reconstruction systems using structured-light images and multi-view point data. The focus then shifts to VSLAM robotics systems in Chapter 4, where a visual data encoding framework is integrated into a monocular ORB-SLAM system to reduce communication overhead while preserving localization and mapping performance. Subsequently, Chapter 5 investigates collaborative multi-robot navigation through an edge-assisted ORB-SLAM architecture, emphasizing efficient communication, global map fusion, and trajectory estimation. To demonstrate practical deployment, Chapter 6 presents the hardware-software co-design and FPGA implementation of the proposed encoding pipeline for resource-constrained robotic platforms. Finally, Chapter 7 summarizes the main findings and contributions of the dissertation and discusses potential directions for future research in visual data encoding, edge-assisted robotics perception, and embedded machine vision systems. Supplementary figures, tables, and additional experimental results that support the analysis presented throughout the dissertation are provided in the appendices. These materials have been organized according to the corresponding chapters to improve readability while maintaining compliance with the dissertation page-limit requirements.

Chapter 2

Baseline DFT-Domain Compression Method for Spatial Perception

2.1 Literature Background and Motivation

Digital images inherently contain redundant information that can be reduced without significantly degrading perceptual quality. The degree of this redundancy is not fixed. It depends on factors such as the application requirements, the intended use of the image, and the expected reconstruction quality [32, 33]. When images are viewed primarily by humans, a relatively higher level of data reduction can be tolerated due to the characteristics of the Human Visual System (HVS), which is less sensitive to certain types of information loss [34, 35]. A major source of redundancy lies in the similarity between neighboring pixels, where adjacent pixels often vary only slightly. This allows some information to be removed or approximated without noticeably degrading image quality. In this context, Low-Frequency Components (LFC) carry most of the essential visual information, whereas High-Frequency Components (HFC) mainly represent edges, textures, fine details, or noise. As a result, many compression techniques aim to preserve low-frequency information while reducing the contribution of high-frequency components in order to achieve efficient data reduction with acceptable quality [36, 37].

The most widely used transforms in image compression are the Discrete Cosine Transform (DCT) and the Discrete Wavelet Transform (DWT), which form the basis of the JPEG and JPEG2000 standards, respectively. The DCT is typically applied

to small, fixed-size blocks (e.g. 8×8), whereas the DWT operates on larger image regions or entire images [38, 39]. In contrast, the Discrete Fourier Transform (DFT) is less commonly employed in practical compression standards. A key limitation of the DFT is that it produces both real and imaginary components for each coefficient, which increases the amount of data that must be processed and stored. As a result, direct DFT-based compression is generally less efficient than DCT- or DWT-based approaches, particularly in terms of energy compaction. Despite this drawback, the DFT remains a fundamental tool in image processing, as it provides a complete frequency-domain representation and enables a clear separation between low- and high-frequency components. In this chapter, the DFT is used as the initial transform in the proposed compression pipeline. This choice is motivated not only by its ability to represent frequency information, but also by its strong support in widely used implementation platforms, such as MATLAB Simulink and AMD Vivado and Vitis, including their Intellectual Property (IP) Library cores. Additionally, it can be implemented efficiently using the Fast Fourier Transform (FFT) scheme. Note that, while these tools provide extensive support for signal processing operations, they do not offer direct standard IP cores for full DCT- or DWT-based compression pipelines. This makes the DFT a more practical and flexible choice for hardware implementation,

particularly in hardware-software co-design and FPGA-based deployment.

To address the inherent limitations of the DFT for compression, the present work introduces three original contributions described in Section 2.2, that together form a novel compression framework:

- The application of 2D DFT as the transform domain within the matrix reduction framework (extending the existing DCT/DWT literature to a new transform domain).
- The replacement of the full probability lookup table (Limited-Data) introduced in [40, 41] to only two bounding parameters, minimum and maximum, which reduces header overhead and increases the compression ratio.

- A novel zero-elimination encoding technique is proposed to compress the high-frequency coefficient matrices.

This combination enables the overall framework to achieve higher compression ratios while maintaining acceptable reconstruction quality, making it competitive with established standards such as JPEG. The primary objective of matrix reduction is to reduce the high-frequency components to one-third of their original size by merging every three data elements into a single representative value. Although this process increases redundancy in the transformed coefficients, it significantly reduces the data volume. However, a known limitation of this approach is the generation of a relatively large probability dataset, referred to as Limited-Data [42, 43, 44], which must be stored as indices for accurate reconstruction during decompression. The combination of DCT with matrix reduction achieved compression ratios exceeding 98% for structured-light images and approximately 95% for conventional images . In the present study, the use of DFT aims not only to demonstrate the general applicability of the matrix reduction technique with different transform domains but also to investigate the potential of DFT as a viable alternative for image compression. To further improve efficiency, the proposed method introduces an optimized representation of the probability data by reducing the lookup table to only two parameters, minimum and maximum values, instead of storing the complete Limited-Data table. This modification significantly reduces the overhead associated with the compressed file header, thereby increasing the overall compression ratio.

The proposed compression algorithm is evaluated using standard performance metrics, including the Compression Ratio (CR), the Root Mean Square Error (RMSE) and the Peak Signal-to-Noise Ratio (PSNR). Experimental results demonstrate that the proposed method achieves competitive compression performance while maintaining acceptable reconstruction quality and compares favorably with widely used techniques such as JPEG.

2.2 The Proposed Compression Algorithm

The proposed compression method is illustrated in Figure 2.1. Initially, an original image is subdivided into non-overlapping blocks of size $m \times n$ pixels starting at the top left corner of the image. The following subsections explain the compression procedure sequentially, starting with the processing of each block and ending with the generation of the final compressed bit stream.

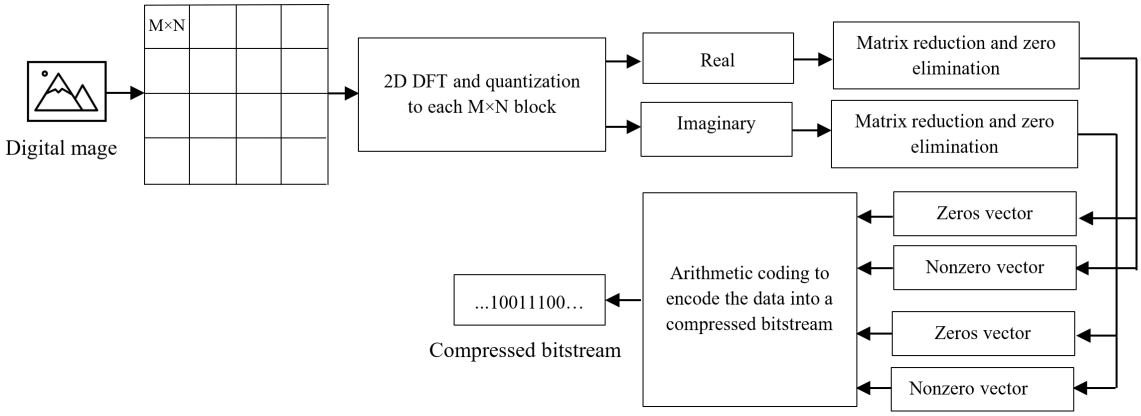


Figure 2.1: The proposed compression method.

2.2.1 Discrete Fourier Transform and Quantization

The Fourier transform is widely regarded as one of the fundamental mathematical tools in signal and image processing because it establishes a link between the spatial and frequency domains. By applying the DFT, and its computationally efficient implementation, the FFT, image data can be represented and analyzed in terms of frequency components. This frequency-domain representation enables various processing operations, including filtering, enhancement, and spectral analysis, to be performed more effectively [45]. The 2D DFT can be calculated mathematically according to the following equation:

$$F[u,v] = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f[m,n] e^{-i2\pi\left(\frac{um}{M}+\frac{vn}{N}\right)}, \quad u = 0,\ldots,M-1,\ v = 0,\ldots,N-1 \quad (2.1)$$

In this representation, $F[u,v]$ denotes the frequency-domain spectrum of the image,

where each pair of coordinates (u, v) corresponds to a particular spatial frequency component. The term $f[m, n]$ represents the image intensity at spatial coordinates (m, n), while M and N denote the image or block dimensions along the corresponding spatial directions, respectively, and i is the imaginary unit. Typically, low-frequency components are concentrated near the center of the spectrum and mainly describe gradual intensity variations and the overall structure of the image. In contrast, high-frequency components are associated with rapid intensity changes, including edges, textures, and fine image details [46].

The original image $f[m, n]$ can be reconstructed from its frequency-domain representation using the 2D Inverse Discrete Fourier Transform (IDFT), defined as [47]:

$$f[m,n] = \frac{1}{MN} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F[u,v] e^{i2\pi\left(\frac{um}{M} + \frac{vn}{N}\right)}, \quad m = 0, \ldots, M-1,\ n = 0, \ldots, N-1 \tag{2.2}$$

The DFT is applied to each image block independently to obtain its frequency-domain representation consisting of real and imaginary components. The matrix reduction algorithm is then applied to each component, followed by a zero-elimination step. Finally, the resulting vectors are subjected to arithmetic coding to generate the final compressed data stream. To illustrate the process for each block $M \times N$ (M = N = 4) in the original image, we represent a block of 4×4 as shown in Figure 2.2. The DFT is applied to each block, followed by a uniform quantization of the resulting real and imaginary components. The quantization process involves dividing each coefficient by a quantization factor (Q) and rounding the results to the nearest integers. This process increases the probability of obtaining high-frequency coefficients with small magnitudes or zero values, thereby reducing the number of bits required to represent these coefficients and consequently improving the compression ratio. Figure 2.3 illustrates the quantization and rounding steps.

It should be noted that the uniform quantization for real and imaginary components is selected heuristically. Up to this point, two matrices (Qr and Qi) are generated per block, representing the quantized real and imaginary components, respectively. Regarding the real part, all low coefficient values (i.e. the DC values) are detached

and saved into a new matrix for Low Frequency Coefficients (LFC-matrix), and it is substituted with a zero value in the quantized matrix. Note that DC values are only found in the real parts, which contribute significantly to the main details and characteristics of the image. The generated LFC-Matrix consists of all the DC coefficients extracted from the entire image and is relatively small compared to the High Frequency Coefficients (HFC-Matrix) which can be represented with a few bytes. Figure 2.4 illustrates the content of the three generated matrices.

Matrix = 4x4	15	18	26	48	Convert matrix to frequency domain by DFT	Real = (Matrix)
	14	19	27	31		
	16	21	32	31		
	16	25	35	32		

Real = (Matrix)	406	-59	-44	-59
	7	10	-21	0
	8	5	-14	5
	7	0	-21	10
Imaginary = (Matrix)	0	59	0	-59
	17	14	3	-26
	0	21	0	-21
	-17	26	-3	-14

Figure 2.2: Application of the DFT to a 4×4 image block.

Real =	406	-59	-44	-59	After quantization (real and imaginary are divided by Q of "20")	Qr=	20	-3	-2	-3
	7	10	-21	0			0	1	-1	0
	8	5	-14	5			0	0	-1	0
	7	0	-21	10			0	0	-1	1
Imaginary=	0	59	0	-59		Qi=	0	3	0	-3
	17	14	3	-26			1	1	0	-1
	0	21	0	-21			0	1	0	-1
	-17	26	-3	-14			-1	1	0	-1

Figure 2.3: Quantization and rounding of real and imaginary components

LFC-Matrix = [20,... etc.] **(DC value for each block 4x4 saved in LFC-Matrix)**

HFC_{Real} = [0 -3 -2 -3 0 1 -1 0 0 0 -1 0 0 0 -1 1]

HFC_{Imag} = [0 3 0 -3 1 1 0 -1 0 1 0 -1 -1 1 0 -1]

Figure 2.4: Decomposition of the 4×4 DFT coefficient matrix into real and imaginary matrices.

2.2.2 Coefficient Reduction Algorithm

Since the HFC-Matrices are considerably larger than the LFC-matrix, the HFC-matrices corresponding to both the real and imaginary components need to be fur-

ther reduced to achieve a reasonable compression ratio. The coefficient reduction algorithm is employed to reduce the size of HFC-matrices by combining every three coefficients into a single equivalent value, which can subsequently be reconstructed to recover the original values during the decompression phase. The reduction is performed on every three consecutive coefficients using predefined Random-Weight-Values. Each coefficient is multiplied by a distinct random number (encoding key) and then their summation is calculated to generate a single encoded value, this concept can be formulated in Equation 2.3.

$$E_i = \sum_{j=1}^{3} C_j K_j \tag{2.3}$$

where E_i is the ith encoded value, C_j is the jth input coefficient, and K_j is the corresponding encoding key. The index i ranges from 1 to $MN/3$, where M and N denote the number of rows and columns of the input coefficient matrix, respectively, and MN is zero-padded to the nearest multiple of 3, if necessary. The index j ranges from 1 to 3, since three consecutive coefficients are combined to generate a single encoded value. Figure 2.5 illustrates the matrix reduction process applied to an $M \times N$ matrix.

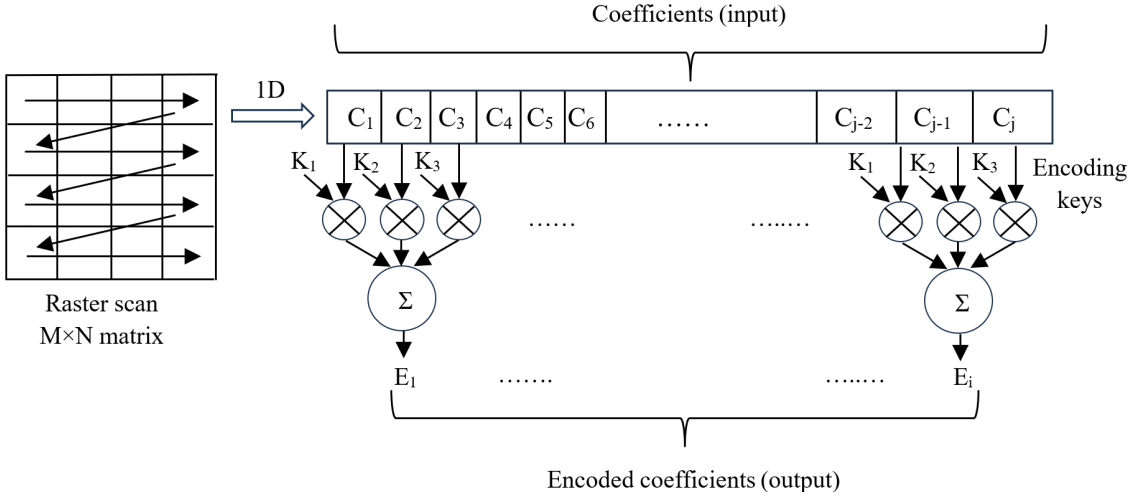


Figure 2.5: Illustration of the matrix reduction method for an $M \times N$ matrix.

During the decompression phase, a search algorithm is used to recover the three original coefficients that are represented by each encoded value. The range of possible

coefficient combinations in the traditional matrix reduction algorithm is stored in a predefined array to facilitate the search process during decoding. These values are added in the header information to be used later in the decompression stage. The decoding process is defined in equation 2.4.

$$(\hat{C}_{i,1}, \hat{C}_{i,2}, \hat{C}_{i,3}) = \left\{ (C_1, C_2, C_3) \in [C_{\min}, C_{\max}] \;\middle|\; E_i = \sum_{j=1}^{3} C_j K_j \right\}, \qquad (2.4)$$

where $(\hat{C}_{i,1}, \hat{C}_{i,2}, \hat{C}_{i,3})$ denotes the reconstructed coefficient triplet corresponding to E_i, and $C_{\min}$ and $C_{\max}$ define the bounded search range used during decompression. However, complex images may generate considerably large arrays, which can adversely impair efficient compression and increase computational demanding. For this reason, this work proposes using the DFT together with a reduced limited search space (a search space contains only two values bounded by $[C_{\min}, C_{\max}]$). Therefore, the minimum and maximum values of the $M \times N$ block are stored in the compressed header. The idea behind this bounding is to limit the range of values required to recover the original three values that produce the encoded value, hence increasing the speed of the search algorithm in the decompression stage.

2.2.3 Encoding of Zero and Nonzero values

After the matrix reduction algorithm is applied, the produced HFC-Matrix for both real and imaginary parts are examined and it is possible to see a high probability in the number of zero values than any other values in the matrix. Therefore, separating zero from nonzero values will remove redundant data and hence increase the efficiency of the arithmetic coding compression. The implementation of this method is by isolating all zero values from the matrix while preserving all nonzero values in a new array called the value matrix. The total number of zeros removed between each nonzero value in the HFC-Matrix is counted during the process. A new array called zero matrix is then created in which a zero value is appended whenever a nonzero value appears at the same index in the original HFC-Matrix followed by an integer that represents the total number of zeros between any two nonzero values. Figure 2.6 demonstrates the process of separating zeros and nonzero values.

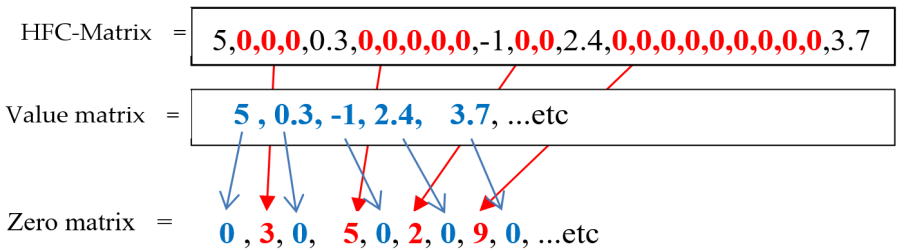


Figure 2.6: Separating zero and nonzero values from each HFC matrix into a zero matrix and a value matrix.

The zero entries in the Zero Matrix indicate the positions of the actual nonzero values in the original matrix. Likewise, the integer entries indicate the number of consecutive zeros that follow each nonzero value. Finally, the two matrices are ready for compression using a lossless coding method, which in this case is arithmetic coding. It is important to note that the proposed method described above is also applied to the LFC-Matrix, which contains the low frequency coefficients values of the real part. Up to this point, the Value-Matrix and Zero-Matrix in our case are considered headers and used in the decompression process to regenerate the original HFC and LFC matrices.

2.3 The Decompression Algorithm

The decompression algorithm performs the inverse operations of the compression process to recover the original image data. The decompression process starts with decoding the LFC-Matrix, Value Matrix, and Zero Matrix using arithmetic decoding. Subsequently, a unified array is reconstructed from the Value Matrix and the Zero Matrix. The reconstructed unified array is then used to regenerate the original HFC-Matrix for both the real and imaginary components. The unified array is reconstructed by reversing the zero-separation process performed during compression. Specifically, the nonzero coefficients stored in the Value Matrix are sequentially inserted into their original positions, whereas the corresponding integer values stored in the Zero Matrix are used to restore the appropriate number of zero coefficients between successive nonzero values.

A sequential search algorithm is used to reconstruct the original coefficients from

each encoded value. The search process iteratively examines all possible coefficient combinations within a bounded search range defined by the minimum and maximum coefficient values preserved during the compression stage. Figure 2.7 illustrates the sequential searching process, where E_i denotes the ith encoded value obtained from the compressed data stream, while K_1, K_2, and K_3 denote the encoding keys (random weight values) used during the coefficient reduction stage. The objective of the search process is to recover the reconstructed coefficient triplet $(\hat{C}_1, \hat{C}_2, \hat{C}_3)$ corresponding to each encoded value. The minimum and maximum values are retrieved from the compressed header and are used to limit the search space, thereby accelerating the reconstruction of the HFC matrices for both the real and imaginary components. Finally, inverse quantization and inverse DFT are applied to each part to reconstruct the compressed digital image. Figure 2.8 illustrates the decompression steps.

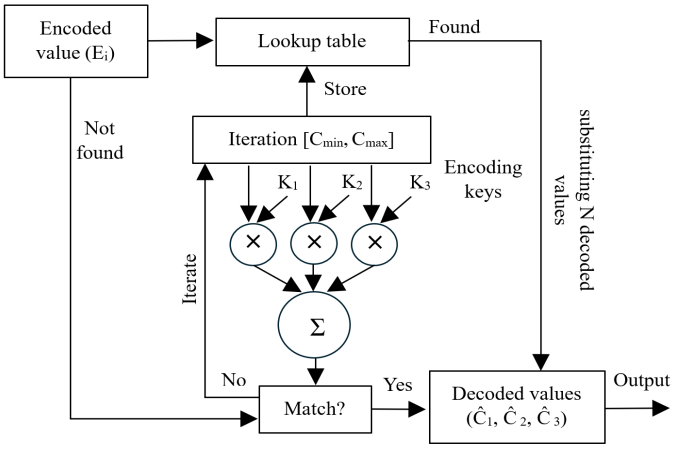


Figure 2.7: Sequential search mechanism used during decompression.

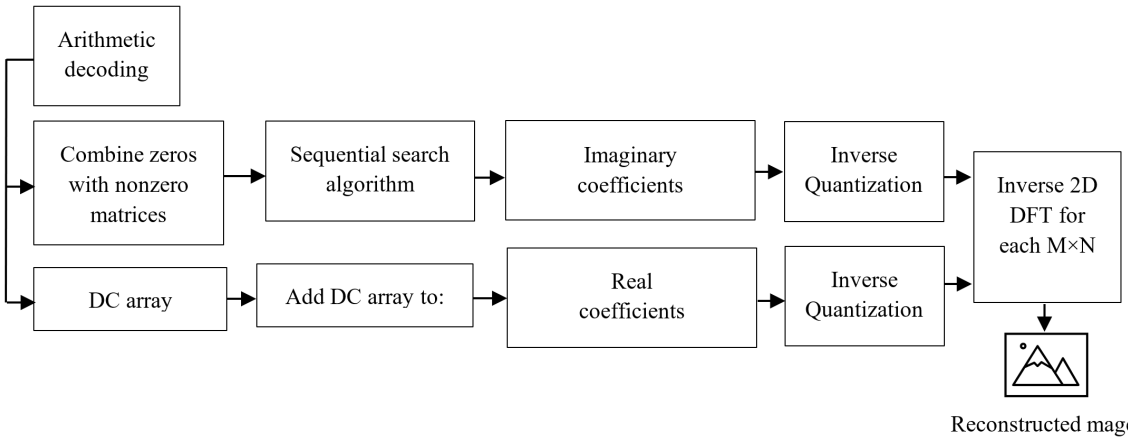


Figure 2.8: The proposed decompression steps.

2.4 Experimental results

The experimental results presented in this section demonstrate the efficacy of the proposed compression method. The proposed method is implemented in MATLAB R2014a running on a quad core Intel i7-3740QM processor. For clarity, the results are divided into two parts:

2.4.1 Results for grayscale and color images

The method is applied to general 2D images of different sizes, and the quality of the reconstructed images is assessed using RMSE and PSNR. Tables 2.1 and 2.2 show the performance results by applying the proposed compression and decompression method to four selected images (two grayscale and two colored) whose details are shown in Figures A.1 and A.2.

Table 2.1: Compression results for grayscale images.

Image	**Original size [MB]**	**Q**	**Compressed size [KB]**	**bpp**	**RMSE**	**PSNR [dB]**
Lion	1.37	25	201	0.143	2.5	44.1
		60	108.4	0.077	5.0	41.1
		100	71.4	0.050	8.1	39.0
Apples	1.37	10	228	0.162	1.2	47.3
		30	91.7	0.065	2.6	43.9
		60	47.8	0.034	4.6	41.5

Table 2.2: Compression results for color images.

Image	**Original size [MB]**	**Q (R,G,B)**	**Compressed size [KB]**	**bpp**	**RMSE**	**PSNR [dB]**
Boeing 777	6.15	10	437.4	0.069	2.1	44.9
		25	182.8	0.029	3.9	42.2
Baghdad	8.58	25	426.3	0.097	4.4	41.6
		35	309.8	0.070	5.6	40.6

2.4.2 Results for Structured-light Images and 3D Surfaces

The proposed method is also applied to structured-light images that are used to reconstruct 3D surfaces. A method for 3D surface mesh reconstruction from structured-light images developed by Rodrigues et al. [48, 49] within the Group of Modeling and Pattern Recognition (GMPR) at Sheffield Hallam University is used. The working principle of the 3D mesh scanner is that the scene is illuminated with a stripe pattern whose 2D image is then captured by a camera. The relationship between the light source and the camera determines the 3D position of the surface along the stripe pattern. The scanner, shown together with the resulting images in Figure 2.9, converts a surface into a 3D mesh in a few milliseconds using a single 2D image [50, 51].

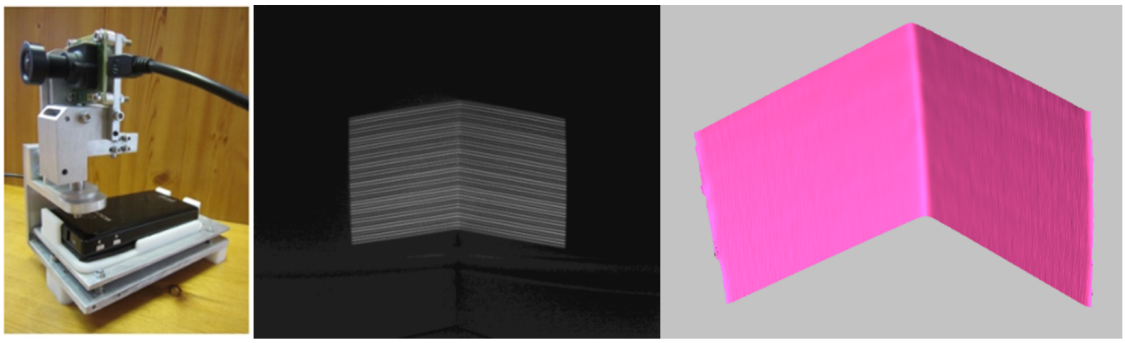


Figure 2.9: (a) The 3D Scanner developed by the GMPR group, (b) a 2D image captured by the scanner, and (c) the 3D surface patch reconstructed from the 2D image.

The significance of using efficient compression of structured-light images is that, if the compression method is lossy and introduces substantial distortion or noise, the 3D reconstruction algorithm may fail to reconstruct the 3D surface accurately or may produce a surface with noticeable artifacts. Consequently, the reconstructed 3D surface may exhibit defects and degradation in the affected regions. Conversely, if the 2D compression and decompression methods preserve sufficient reconstructed image quality, the 3D surface can be reconstructed accurately, with no visually significant differences between the reconstruction obtained from the original images and that obtained from the decompressed images. Figure 2.9 (left) depicts the GMPR

scanner together with an image captured by the camera (middle), which is then converted into a 3D surface and visualized (right). Note that only the portions of the image containing patterns (stripes) can be converted into 3D, other parts of the image are ignored by the 3D reconstruction algorithm. The original images used in this investigation (Corner, Face-1, and Face-2) are shown in Figure 2.10. The three images were compressed using the proposed method, and their compressed sizes, together with the corresponding RMSE and PSNR values, are presented in Table 2.3. After decompression, the images are subjected to 3D reconstruction using the GMPR method and compared to the 3D reconstruction of the original images. The reconstructed 3D surfaces are shown in Figures A.3, A.4, and A.5.



Corner (1280x1024) Face1 (1392x1040) Face2 (1392x1040)

Figure 2.10: Original 2D images with different sizes used by the proposed compression method.

Table 2.3: Compressed 2D structured-light images using the proposed method.

Image	**Original Size[MB]**	**Quantization**	**Compressed size [KB]**	**bpp**	**RMSE**	**PSNR [dB]**
Corner	1.25	60	35.4	0.027	4.7	41.4
		100	17.8	0.013	15.5	36.2
Face-1	1.37	100	34.0	0.024	8.4	38.8
		160	18.1	0.012	11.5	37.5
Face-2	1.37	50	46.2	0.032	6.7	39.8
		150	20.1	0.014	9.9	38.1

2.5 Discussion and Comparative Analysis

The literature provides very little research of image compression using DFT alone. The main reason is that the DFT produces two sets of frequency domain coefficients, namely the real and imaginary components. Preserving both components for accurate image reconstruction significantly increases the amount of data that must be stored or transmitted, which limits the achievable compression ratio. Consequently, using DFT alone cannot achieve comparable compression ratios while maintaining practical storage requirements.

In contrast, combining DFT with the matrix reduction algorithm and zero/nonzero coding helps overcome these limitations. The additional processing stages reduce redundancy and decrease the amount of encoded data while preserving an acceptable level of reconstructed image quality. As shown in Table A.1, the proposed method achieves substantially smaller compressed sizes and lower bits per pixel (bpp) values compared to using DFT alone. Although the proposed approach produces higher RMSE values and lower PSNR values in some cases due to the aggressive reduction of high-frequency information, the reconstructed images still maintain high visual quality while providing significantly higher compression efficiency.

The experimental results also indicate that the proposed method is capable of achieving compression ratios of up to 98%, while DFT alone cannot provide comparable compression performance at a similar level of reconstructed image quality. Table A.2 shows that the proposed method achieves competitive RMSE and PSNR values despite the loss of some fine image details at high compression ratios. Nevertheless, the reconstructed images still maintain an acceptable level of visual quality. In addition, the proposed method is compared with the JPEG standard [52, 53, 54], which remains one of the most widely used techniques in image and video compression and many areas of digital image processing [55]. The JPEG relies on the DCT and Huffman coding, while the proposed method employs DFT in combination with matrix reduction and zero/nonzero coding. Therefore, the comparison provides a useful evaluation of the proposed approach against a well-established conventional

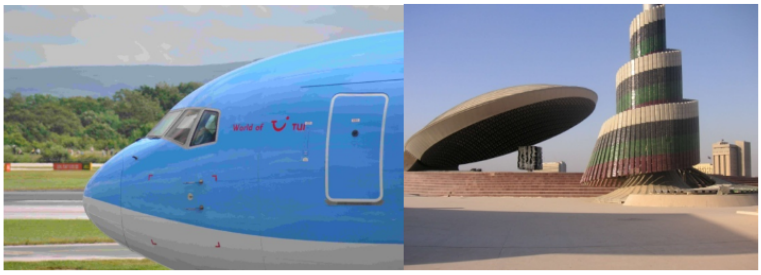
compression framework. Table A.2 presents the analytical comparison between the two methods.

The results indicate that the proposed method achieves better compression performance than JPEG for structured-light images. For conventional images, both methods can be considered relatively comparable, although the reconstructed image quality varies between the two approaches. Figures 2.11, 2.12, and A.6 present visual comparisons between the proposed method and the JPEG standard for the images listed in Tables A.1 and A.2.



(a) Compressed Size=56 KB (b) Compressed Size=48 KB

Figure 2.11: Decompressed greyscale images using JPEG, the quality of the decompressed images varies compared with the proposed approach according to RMSE and PSNR.



(a) Compressed Size=210 KB (b) Compressed Size= 279 KB

Figure 2.12: Decompressed color images by JPEG, the decompressed image (Boeing) has lower quality compared with the proposed approach according to RMSE and PSNR. However, the approach could not reach the JPEG compression level for Baghdad's image on the right.

Concerning the compression of structured-light images for 3D mesh reconstruction, the comparison between the proposed method with JPEG standard demonstrates

the strong potential of the proposed approach, as depicted in Figures 2.10, A.3, A.4, and A.5. Trying to compress the same images using JPEG and then using the decompressed images to generate the 3D mesh clearly reveals the problems and limitations of JPEG for this types of applications. These limitations are further demonstrated in Figure A.6, showing the 3D mesh reconstruction results obtained from two structured-light images compressed using JPEG. These limitations are further demonstrated in Figure A.6, which presents the reconstructed results obtained from two structured-light images compressed using JPEG.

The comparative analysis also considers previous work on matrix reduction based on DWT and DCT transforms [39, 43, 44], where compression and encryption were performed simultaneously. However, the relatively high computational complexity of both the compression and decompression stages was identified as one of the main limitations of the previous approaches. Table A.3 compares the decompression time of the conventional matrix reduction algorithm [39] with the proposed method. The main advantages of the proposed approach over the previous methods can be summarized as follows:

- The complexity of the decompression steps is reduced in the proposed approach. This is evident from the execution times reported in Table A.3, where the proposed approach runs faster than the conventional approach on the same hardware.
- The header file information of the proposed method is smaller than that of the conventional searching approach, leading to increased compression ratios.

It is important to emphasize the novelty of the proposed approach, namely the reduced number of processing steps during the decompression stage and the smaller amount of header information. Together, these advantages contribute to faster decompression of highly compressed image data. Table A.4 shows that the proposed image compression method achieves significantly higher compression ratios while maintaining competitive RMSE and PSNR values for both conventional and structured-light images.

2.6 Thesis I.

I introduced a novel image compression algorithm based on the 2D Discrete Fourier Transform (2D DFT) with an enhanced coefficient reduction algorithm and a zero elimination technique, achieving high compression ratios while preserving image quality.

Supporting Statement:

This contribution addresses a fundamental limitation of DFT-based compression, namely the difficulty of efficiently compressing real and imaginary components without degrading the fidelity of the reconstruction. By integrating DFT with the proposed minimization strategy, the method overcomes this constraint and achieves a high-compression efficiency while preserving reconstruction quality, thereby outperforming the traditional JPEG standard.

Related Publications: [P15], [P11], [P12].

2.7 Chapter Findings

This chapter presented a novel 2D image compression method for grayscale, color, and structured-light images used in 3D reconstruction applications. The proposed method combines DFT, matrix reduction algorithm, and zero/non zero coding techniques to achieve high compression efficiency while maintaining acceptable reconstruction quality.

The original scientific contribution of the method can be summarized as follows:

- The input image is divided into non-overlapping 4×4 blocks, and the DFT is applied independently to each block. The resulting real and imaginary components are quantized separately. In addition, the low frequency Coefficients (DC values) are extracted and stored in a dedicated matrix, while the remaining coefficients represent the high frequency Coefficients.
- The matrix reduction algorithm reduces the size of the high frequency coef-

ficients to one-third of their original size, thereby significantly increasing the achievable compression ratio.

- The conventional probability lookup table used during decompression is replaced by two bounded search parameters (minimum and maximum values). This modification reduces header overhead, improves reconstruction speed, and enhances overall compression efficiency.

Experimental results demonstrate that the proposed method achieves competitive image quality while providing higher compression ratios, particularly for structured-light images used in 3D surface reconstruction. Furthermore, the reconstructed 3D surfaces exhibit minimal visual degradation despite aggressive compression. Comparative analysis with JPEG and previous matrix reduction approaches indicates that the proposed method provides improved compression efficiency and competitive PSNR and RMSE performance, especially for structured-light applications. The proposed compression framework demonstrates strong potential for applications that require high compression ratios while preserving sufficient reconstruction quality for both conventional imaging and 3D reconstruction systems.

Chapter 3

Beyond JPEG Limitations: Improved Image Compression for 3D Reconstruction

3.1 Literature Background and Introduction

Recently, the demand for geometry modeling has grown for a wide range of applications such as aerospace models, medicine, computer science, automotive Computer-Aided Design (CAD) datasets, engineering, virtual reality, entertainment, and so on. Modeling 3D objects can be classified into two primary categories namely surface modeling and solid modeling. Surface modeling is the branch in which we can represent and deal with 2D surfaces embedded in 3D space. A popular polynomial discrete representation method for 3D surfaces is the mesh generation from scanned images. Its popularity comes from their algorithmic simplicity, ease of calculation on Graphics Processing Unit (GPU), and rendering and displaying efficiency [56]. A 3D mesh consists of three basic elements: edges, vertices, and faces. The edges basically are lines connecting vertices. Faces are closed surfaces created by edges. Vertices are formed by connecting neighboring edges. In regular meshes, all faces have the same size, and also all vertices have the same valence. Among many representation methods, triangular meshes may be considered as the most common method due to their simple and efficient modeling [57]. Current 3D higher cost storage size application devices have high technology graphics cards for rendering 3D meshes with the aid of

special software for editing and visualizing data. Most of these applications need to save as well as transmit, share, and exchange 3D data over networked environments and the Internet. On the other hand, the large size of 3D mesh data requires higher storage requirements and imposes hard limits on bandwidth due to the large amount of data [58]. This calls for active research on more efficient 3D data compression and reconstruction algorithms to satisfy the demands of 3D modeling. In the literature [37, 39, 40, 44, 58, 59, 60] many different techniques have been proposed for 3D mesh data compression and reconstruction based on high degree polynomial interpolation or partial differential equations. An approach confines the concept that the actual geometry of a 3D solid object is the 2D surface. In other words, the objective is to compress the 2D surface image at a high compression ratio with minimal information loss while preserving the quality of the subsequent 3D reconstruction. In this way, we can reduce the 3D computational complexity, transmitting time, bandwidth limitations, and storage requirements [41].

The focus of this work is on compression of 3D data reconstructed from multi-view images or from a single structured-light image. Because 3D structures are obtained from images, we naturally focus on quality image compression. Structured-light images contain patterns of light and shadows on the surface of the 3D object that are captured by the sensor at very high resolution, allowing accurate 3D object reconstruction. In relation to the method proposed in this chapter, previous work on compressing 2D image data tailored for 3D reconstruction from structured-light images are first proposed in [61]. Specifically, a single level DWT is applied on the input image, and then a DCT on the LL sub-band producing the DC and AC components. A single level DWT is applied again to the DC component, and then the resulted LL2 sub-band is transformed again by DCT. The AC coefficients are reduced using the reduction algorithm, resulting in a compression ratio of up to 98%. Similarly, the study presented in [43] developed a decompression-stage technique by proposing an algorithm called the Limited-Data Sequential Search Algorithm, in which similar transformations are applied to the data blocks, followed by arithmetic coding during the compression stage. The results show better image quality at higher compression ratios than both JPEG and JPEG2000. However, the overall complexity

of these algorithms increases the computational time at both the compression and decompression stages. In [44], a two-level DWT followed by a DCT was proposed to generate DC and AC components. A second DCT was then applied to the AC components, producing two arrays: zeros and nonzero arrays. Subsequently, a matrix reduction size algorithm was applied to the AC coefficients and the remaining high frequencies followed by arithmetic coding. A novel fast match search decompression was proposed to reconstruct all high-frequency components by calculating all compressed data probabilities using a binary search algorithm to estimate the data with the aid of a lookup table. The results showed higher compression ratios than JPEG and JPEG2000 with better visual image quality and accurate 3D reconstructing. On the other hand, the complexity of the FMS-based binary search algorithm led to a remarkable increase in the execution time compared with previous works. In [44], a new method for compression based on two transforms is introduced, a 1D-DCT applied to each row and 1D-DST (Discrete Sine Transform) applied to each column of the input image followed by quantization for the high frequencies and then coded by matrix minimization algorithm followed by arithmetic coding. A binary search based on fast matching search algorithm was used at decompression yielding up to 99% compression ratio.

The literature clearly indicates that effective image compression techniques commonly rely on combinations of multiple transforms, including the DWT, DCT, and DST, together with algorithms such as matrix reduction and sequential or binary data search. These approaches aim to achieve higher compression ratios while preserving accurate surface reconstruction quality. However, the integration of multiple transforms and optimization algorithms significantly increases computational complexity and processing time. The main original contribution of this chapter, presented in Section 3.2, is a novel lossy encoding method based on a single-level DCT pipeline incorporating an optimized quantization matrix, DC differencing, zero-elimination, and arithmetic coding. Building upon the existing DCT-based literature, the proposed approach addresses several limitations of previous studies when applied to the compression of multi-view images and structured-light images used for 3D surface reconstruction.

3.2 The Proposed Compression Pipeline

The proposed compression pipeline is illustrated in Figure 3.1. As can be shown, the image is first preprocessed by subdividing it into non-overlapping $m \times n$ blocks. For a direct comparison with JPEG we use the standard block size of 8×8 pixels starting from the upper left corner of an image. The image may be padded with zeros if the image dimensions are not a multiple of eight. Afterward, each block is processed with the steps described in the following subsections.

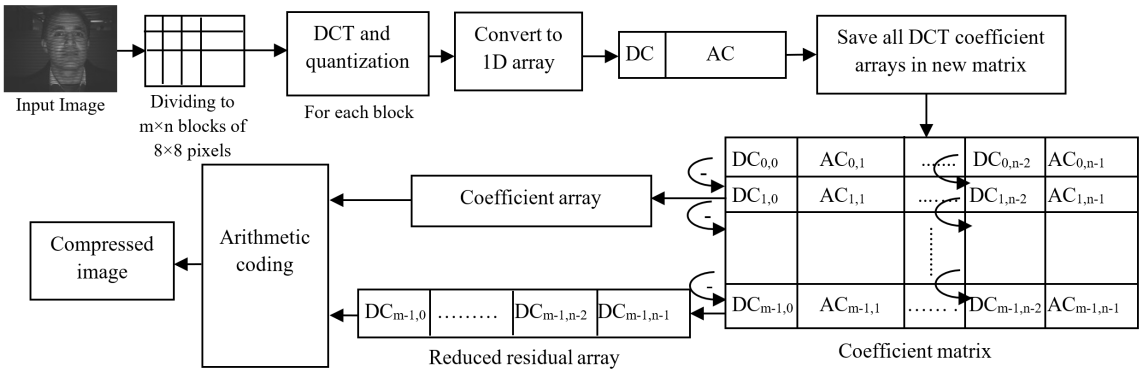


Figure 3.1: The proposed DCT-based compression pipeline.

3.2.1 Discrete Cosine Transform

The DCT is applied to each 8×8 block and saved as a 64 element array. The first element represents a low frequency called the DC component, while the remaining 63 elements are called the AC coefficients which represent the high frequencies. The DCT forward and inverse transforms are given in Equations (3.1) and (3.2), respectively [62].

$$F(u,v) = \frac{2}{\sqrt{MN}} c(u)c(v) \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) \cos\left(\frac{(2x+1)u\pi}{2M}\right) \cos\left(\frac{(2y+1)v\pi}{2N}\right), \tag{3.1}$$

$$f(x,y) = \frac{2}{\sqrt{MN}} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} c(u)c(v)F(u,v) \cos\left(\frac{(2x+1)u\pi}{2M}\right) \cos\left(\frac{(2y+1)v\pi}{2N}\right), \tag{3.2}$$

where $f(x, y)$ denotes the spatial-domain image block, $F(u, v)$ represents the corresponding DCT coefficient at the frequency indices (u, v), x and y denote the spatial-domain indices, and M and N represent the dimensions of the image block. The indices satisfy $0 \leq x, u \leq M-1$ and $0 \leq y, v \leq N-1$. In the proposed method, the DCT is applied to 8×8 image blocks, therefore, $M = N = 8$. The normalization factors $c(u)$ and $c(v)$ are defined in Equation (3.3):

$$c(u) = \begin{cases} \frac{1}{\sqrt{2}}, & u = 0, \\ 1, & u > 0, \end{cases} \qquad c(v) = \begin{cases} \frac{1}{\sqrt{2}}, & v = 0, \\ 1, & v > 0. \end{cases} \tag{3.3}$$

3.2.2 Optimized Quantization

The quantization matrix used in this work was derived from the standard JPEG luminance quantization matrix and subsequently modified to preserve important low-frequency information. Specifically, the DC coefficient, which represents the average intensity of the image block, together with its neighboring low-frequency AC coefficients, which contain the majority of the structural image information, were exempted from quantization by replacing their corresponding quantization values with ones. This modification was introduced to preserve structural image information while maintaining high compression ratios. The selection was further motivated by machine-oriented applications, such as VSLAM, robotics, and 3D reconstruction, where preserving luminance-based structural information is more critical than preserving color fidelity information, since feature extraction algorithms depend mainly on edges, corners, and image details. The quantization matrix Q, applied to each 8×8 DCT coefficient block, is defined in Equation 3.4. The use of ones in the quantization matrix preserves the DC coefficient and the neighboring low-frequency AC coefficients during the quantization process. Since these coefficients contain most of the structurally significant image information, preserving them contributes to maintaining image quality and improving the robustness of feature extraction, particularly at high compression ratios.

$$Q = \begin{bmatrix} 1 & 1 & 1 & 1 & 24 & 40 & 51 & 61 \\ 1 & 1 & 1 & 19 & 26 & 58 & 60 & 55 \\ 1 & 1 & 16 & 24 & 40 & 57 & 69 & 56 \\ 1 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix} \tag{3.4}$$

3.2.3 First Order Difference Processing

In every column, each DC coefficient is replaced by the difference between the DC value of the current row and the DC value of the next row starting from the first row (i = 0) to (i = m-1), that is:

$$\Delta DC_{i,j} = DC_{i,j} - DC_{i+1,j}, \qquad i = 0, \ldots, M-2, \quad j = 0, \ldots, N-1. \tag{3.5}$$

where i and j denote the indices of the block's row and column, respectively, while M and N denote the total numbers of block rows and block columns, respectively. This operation increases the correlation among the DC coefficients and reduces their magnitudes, thereby improving compression efficiency. The last DC coefficient in each column is kept unchanged and stored in a new array called "reduced residual array" (see Figure 3.1), while its original position in the coefficient matrix is replaced by zeros. The coefficient matrix is then transferred to a new array, called the coefficient array, by raster scanning the 64 coefficients of each block row by row. If all the coefficients of a block are zero, they are represented by a single zero in the coefficient array. Otherwise, the 64 coefficient values are stored in the coefficient array, preceded by a value of 1 as a header to indicate that the block contains at least one nonzero coefficient. Figure 3.2 shows the format of the coefficient array. Finally, the coefficient array and the reduced residual array are arithmetically encoded to produce the compressed image file.

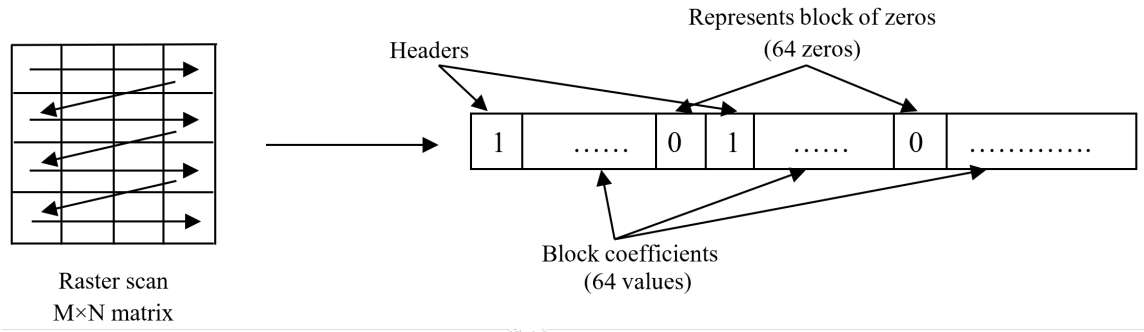


Figure 3.2: The coefficient array format.

3.3 The Proposed Decompression Pipeline

The decompression steps are simply the reverse of the compression algorithm, Figure 3.3 illustrates the steps. First, the compressed image file is arithmetically decoded to obtain the coefficient array and the reduced residual array. The coefficient matrix is re-generated from the coefficient array by scanning and checking the headers (ones) and the zeros in the coefficient array. If a header of "1" is present, the next 64 values are moved to the coefficient matrix in the corresponding 8×8 block index, while if a zero is present, then 64 zeros are substituted.

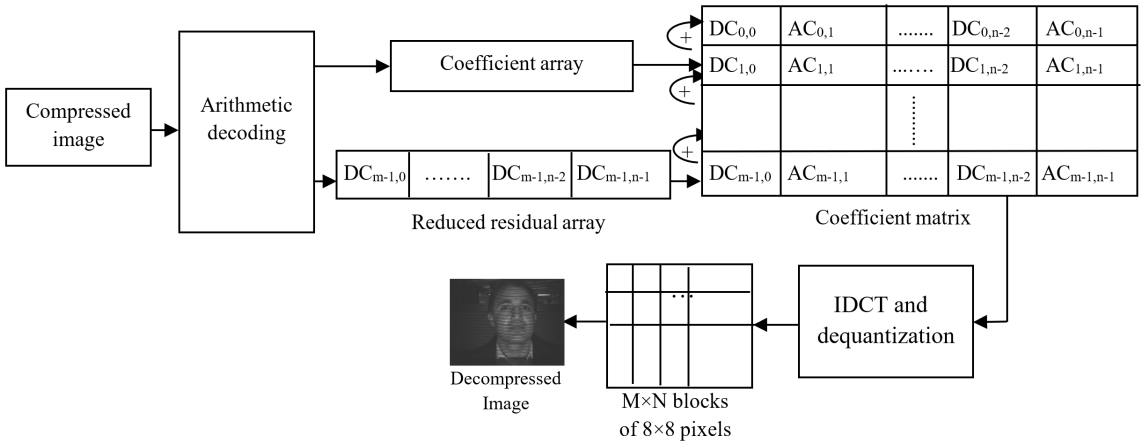


Figure 3.3: The Proposed compression algorithm.

At this stage,, the DC values in the coefficient matrix represent the differences between consecutive DC coefficient within each column. The original DC coefficients are recovered in two steps. First, the values stored in the reduced residual array are restored to the DC positions in the last row. Second, the remaining DC coefficients in each column are recovered by adding each differential DC value to

the reconstructed DC value in the next row, proceeding from the second-to-last row upward to the first row. That is:

$$DC_{i,j} = \Delta DC_{i,j} + DC_{i+1,j}, \qquad i = M-2, \ldots, 0, \quad j = 0, \ldots, N-1. \tag{3.6}$$

where i and j denote the block row and column indices, respectively, while M and N denote the total numbers of block rows and block columns, respectively. Finally, the Inverse Discrete Cosine Transform (IDCT) is applied to each dequantized 8×8 coefficient block to recover the decompressed image.

3.4 Experimental results

The proposed algorithm was experimentally evaluated using structured-light images and multi-view images on an Intel Core i7-3740QM processor, with the method implemented in MATLAB R2014a. Table 3.1 presents the experimental results obtained by compressing three grayscale structured-light images using the proposed method. The corresponding original and decompressed images are shown in Figures B.1, B.2, and B.3.

Table 3.1: Compression performance of the proposed method for structured-light images.

Image	**Original size [MB]**	**Dimension**	**Compressed size [KB]**	**RMSE**	**PSNR [dB]**
Face-1	1.38	1932 × 1040	14	3.3	42.9
Face-2	1.38	1932 × 1040	17.9	4.35	41.7
Face-3	1.38	1932 × 1040	21.1	8.3	38.9

In addition to structured-light images, the proposed method is applied to two different sets of multi-view images for 3D object reconstruction. The 3D objects are reconstructed using 3DF Zephyr software [63] and subsequently converted into 3D meshes using the 3D MeshMixer software from Autodesk [64]. Table 3.2 presents the experimental results obtained by compressing the two sets of multi-view images

using the proposed method. The corresponding original and decompressed images and their reconstructed 3D objects are shown in Figures B.4, B.5, B.6, and B.7.

Table 3.2: 2D multi-view images Compressed using the proposed method at high compression ratios.

Set	**Nr. of frames**	**Total frame size [MB]**	**Total compressed size [MB]**	**Average RMSE**	**Average PSNR [dB]**
Apple	48	114.8	3.1	5.8	41.3
Soft Toy	42	100.38	2.39	11.0	38.2

3.4.1 Comparison with the JPEG standard for structured-light images

Figures B.8, B.9, and B.10 show the 3D mesh reconstruction results obtained from the decompressed structured-light images Face-1, Face-2, and Face-3 using JPEG and the proposed method in direct comparison. The size of the Wavefront Object (.OBJ) file representing each 3D structure is also included for comparison to demonstrate the potential saving in storage space and transmission bandwidth achieved by the proposed method. Table B.1 summarizes the experimental results.

The results clearly show that the 3D surfaces reconstructed from the images compressed using JPEG are substantially degraded, exhibiting artifacts and missing regions across different areas of the surfaces. This degradation occurs because JPEG is unable to perceive sufficient image information for reliable 3D reconstruction at high compression ratios considered, reaching approximately 99% for Face-1 and 98% for Face-2 and Face-3. In contrast, the proposed method achieves higher compression ratios while maintaining better reconstructed image quality and preserving sufficient image details for successful 3D reconstruction.

3.4.2 Comparison with JPEG and 3D Object (.OBJ) File Sizes

The next two Figures, B.11 and B.12, illustrate the 3D mesh surface reconstruction obtained from the two sets of multi-view images compressed using JPEG and the

proposed method. Table B.2 summarizes the experimental results, including the corresponding 3D object file sizes for comparison.

From the figures, it can be observed that JPEG standard is inefficient at preserving the image information required for multi-view 3D reconstruction at high compression ratios. The resulted 3D reconstructions exhibit artifacts and missing surface details as the compression ratio increases. While the proposed method successfully achieves compression ratios of up to 99% while preserving sufficient information for 3D reconstruction, JPEG at approximately 97% produces severely degraded 3D meshes with substantial artifacts and missing surface regions. Finally, Figure B.13 provides further details of the perceptual image quality assessment of the proposed algorithm in comparison with JPEG.

3.5 Thesis II.

I proposed a novel lossy image compression method optimized for 3D mesh reconstruction from structured-light and multi-view images, addressing the limitations of JPEG at high compression ratios. The proposed method improves DCT-based compression by optimizing the quantization matrix and introducing an efficient coefficient management pipeline, including DC differencing, zero-elimination, and arithmetic coding.

Supporting Statement:

The approach achieves up to 99% compression while preserving reconstruction quality, maintaining accurate geometry and surface details, where JPEG fails beyond 97% compression. This contribution provides an efficient solution for storage and transmission in 3D reconstruction pipelines, significantly reducing data requirements without sacrificing geometric fidelity.

Related Publications: [P13], [P14], [P8].

3.6 Chapter Findings

In this chapter, a novel lossy image compression method based on the conventional JPEG standard is proposed for efficient 3D surface reconstruction from structured-light and multi-view images. The proposed approach addresses the limitations of JPEG compression in preserving reconstruction quality at high compression ratios while maintaining lower computational complexity compared with existing methods reported in the literature. Compared with JPEG, the proposed approach better preserved the visual and structural information required for 3D reconstruction, particularly under high-compression conditions where JPEG-compressed images resulted in pronounced artifacts, missing surface regions, or degraded 3D meshes. These findings demonstrate that compression performance for spatial perception applications should be evaluated not only in terms of compressed size and conventional image-quality metrics, but also according to the preservation of information required by the subsequent 3D reconstruction process.

Chapter 4

End-to-End Multi-Level Encoding for Robust Monocular Visual ORB-SLAM

4.1 Introduction

Simultaneous localization and mapping (SLAM) has been extensively studied in the last decade. It allows the estimation of the camera pose of a mobile device and the creation of a map of the surrounding environment concurrently. Recently, Visual SLAM (VSLAM) has become the most widely used state-of-the-art technique to implement the SLAM tasks due to its low cost, compact size, and affordability. However, the intensive computation of VSLAM systems is unsuitable for wide range of mobile devices with limited computational and energy resources. A possible solution is to split its functionality between mobile devices and the edge cloud. This solution showed the necessity for efficient visual data compression methods to be integrated within VSLAM systems. In this chapter, a multi-level encoding method for visual data frame compression integrated within the monocular Oriented FAST and Rotated BRIEF-SLAM (ORB-SLAM) system is proposed. The performance results of the proposed system are compared to corresponding ORB-SLAM systems adopting the most popular classical still image compression standards; the Joint Photographic Experts Group (JPEG) and the advanced version, the JPEG2000, in terms of reconstruction quality, robot's trajectory estimation, and computational complexity.

4.2 Literature Background: Visual Simultaneous Localization and Mapping (Visual SLAM)

In recent years, SLAM has gained great attention in research and industry due to the expansion of robotics applications. SLAM aims to create a map of an anonymous environment while concurrently determining the mobile device's trajectory and position [65]. It can also help reduce estimation errors when the robot is in a previously mapped area. The mobile device can be an indoor robot [66], an Autonomous Vehicle (AV) [67], an Unmanned Autonomous Vehicle (UAV) [68], or an Unmanned Ground Vehicle (UGV) [69]. The SLAM algorithm performs map construction and robot localization based on data obtained from different sensors, such as Light Detection and Ranging (LiDAR) [70], radio signals [71], or visual sensors [72]. The camera-based SLAM system, commonly known as Visual SLAM (VSLAM), is interesting nowadays due to its low cost, rich information gathering, facilitates object detection and tracking, and compact size. VSLAM obtains environmental information from a single camera in monocular VSLAM [73], multiple cameras (stereo VSLAM [74]), or Red, Green, and Blue with Depth (RGB-D) [75] cameras for mapping and localization tasks. VSLAM is traditionally deployed onboard the robot and directly interfaced with a camera that captures non-compressed video frames. However, VSLAM algorithms demand intensive computational resources, which has been a significant challenge in the last decade. Tracking and mapping are the two general tasks to be executed in parallel, which requires high computational costs. Such demanding computing requirements need powerful onboard resources, making them less suitable for mobile devices with severe hardware constraints that decrease their range and usage time. Recent studies have been conducted to provide solutions to reduce weight, save energy, and keep mobile device sizes small by offloading these computationally intensive processing tasks to edge servers and cloud platforms [76, 77]. This practical solution exhibits the need for efficient encoding methods to compress the image data before transmission to decrease bandwidth when performing VSLAM at the edge or in the cloud. Therefore, data compression is necessary to manage and transmit visual information and preserve as much information as possible, especially in high-latency

communication environments. While image compression provides the clear advantages of reducing data load and minimizing bandwidth requirements, it introduces challenges for VSLAM, such as degradation of feature quality, mismatched feature descriptors, and increased computational complexity [78].

Image compression is the technique of reducing the image and video volumes by representing the spatial pixels in different compressed domains using fewer bits while maintaining visual information to a certain extent. Lossy compression is a most popular strategy in machine vision tasks as it provides more data reduction capabilities [79]. However, it may influence image quality adversely, making features more intricate to detect and track during visual SLAM operation and affecting localization accuracy negatively.

There are two different paradigms in the literature regarding cloud-based machine vision applications. The first transmits the image-extracted features of pre-processed raw data on the mobile device. This approach is referred to as Analyze then Compress (ATC). The second is compressing the raw data directly and transmitting it to the server cloud, which is referred to as Compress then Analyze (CTA). These paradigms differ in the order of steps, the amount of data to be transmitted, and the end-to-end latency from the moment of visual data capturing to the destination [78]. Figure 4.1 demonstrates these paradigms.

Many studies have researched the offloading of VSLAM architectures on edge devices and cloud platforms to perform trajectory mapping and localization of mobile robots. The research [80] proposed and implemented a cloud framework for Cooperative Tracking and Mapping C_2TAM. The proposed architecture involves implementing a centralized map on the cloud server while tracking is still running on mobile devices. In this manner, mobile devices send the detected keyframes to the cloud server in a non-compressed format of 640×480 Red, Green and Blue (RGB) images. That means an average throughput of 1 MB/s is required for transmission. Despite the study not evaluating the transmission rate of keyframes, transmitting all raw-sized keyframes demands significant bandwidth. In [81], the Portable Network Graphics (PNG) standard was used to compress the extracted keyframes in collaborative

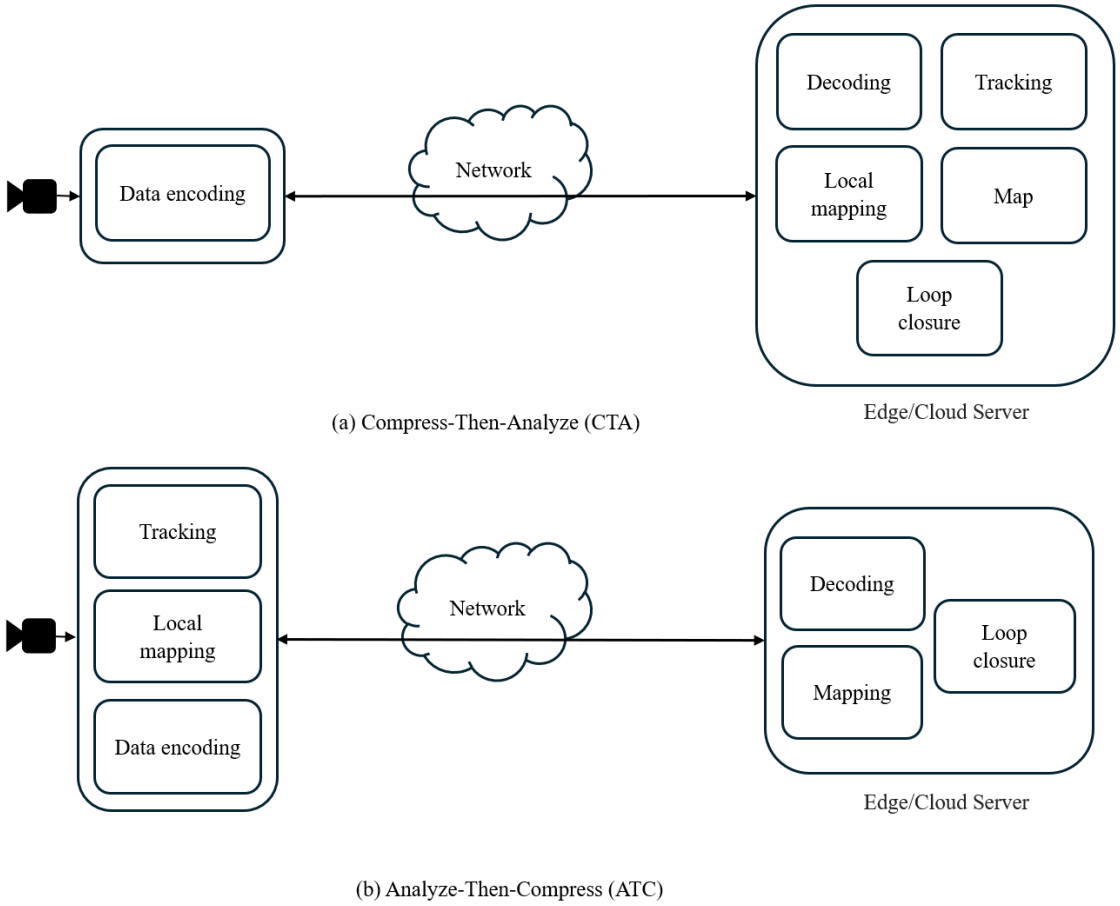


Figure 4.1: Edge assisted VSLAM paradigms: (a) Analyze then Compress (ATC), and (b) Compress then Analyze (CTA).

3D dense visual odometry assisted by the cloud. The cloud integrates the received keyframes, merging them with maps constructed from other robots, then optimizes the pose and relays it to the mobile device. A similar study [82] investigated the influence of lossy data compression on the data size and accuracy. The H264 video codec is adapted to reduce the size of depth data captured by a Kinect camera and 3D LiDAR on a mobile robot. The study aimed to solve the problem of transmitting range data streams over low bandwidth networks. The proposed method proved that highly compressed depth images can still be used in dense mapping algorithms. In contrast, the work in [83] presented a real-time VSLAM for edge agents. The proposed architecture integrates lossy compression for raw data encoding. It demonstrated that the lower bit rates introduced by lossy compression have a negative effect on the feature extraction quality.

In the literature, many compression standards are introduced. These methods can be categorized as classical methods and learning-based methods. Classical methods use predefined mathematical techniques such as transforms and entropy encoding. In contrast, learn-based image compression methods use machine learning techniques to compress data to higher compression ratios. However, classical methods are still a reasonable choice for mobile device implementation due to their lower computational complexity compared to the massive computational demands of learned methods [84, 85, 86].

The studies reviewed above indicate that, although several approaches have investigated visual data compression for VSLAM or edge-assisted robotic perception, none integrates a multi-level, domain-specific encoding framework directly within the ORB-SLAM pipeline while supporting both the CTA and ATC paradigms under bandwidth and computational constraints. This gap motivates the original contribution of this chapter to present a novel multi-level encoding visual data compression method that extends the proposed matrix reduction framework to compress input frames to high compression ratios while preserving the information quality. It examines the feasibility of integrating the proposed method within the CTA and ATC frameworks of ORB-VSLAM robots. The proposed method is compared with equivalent VSLAM architectures employing the JPEG standard and the more advanced JPEG2000 standard. The JPEG is the typical and widely used traditional lossy compression standard, while the JPEG2000 is a more efficient lossy compression standard employing multiresolution capabilities based on the DWT [87]. Inherently, this work also demonstrates the effect of using lossy compression on the system performance in terms of data size, execution time, and trajectory estimation accuracy.

4.2.1 Monocular VSLAM

Visual SLAM estimates the pose and reconstructs a map of the surrounding environment using data collected from visual sensors. It continuously updates the global map to decrease drifts, allowing localization and loop-closing detection. Generally, VSLAM methods are divided into direct and indirect (feature-based) categories. Direct methods work directly on raw pixel data for pose estimation and reduce photo-

metric errors, while indirect methods extract feature points (key points) and match them with subsequent ones.

Visual Features and Keypoint Representation

Features can be considered as digital representations of image content that describe important visual characteristics within an image. The effectiveness of many computer vision systems strongly depends on the selection of an appropriate set of features, motivating extensive research efforts in feature extraction and matching techniques. In contrast to deep learning methods, traditional computer vision approaches rely primarily on handcrafted features, typically represented by keypoints and distinctive image regions used for matching, tracking, and recognition tasks. Common examples of such informative regions include corners, edges, and textured blocks, as illustrated in Figure 4.2.

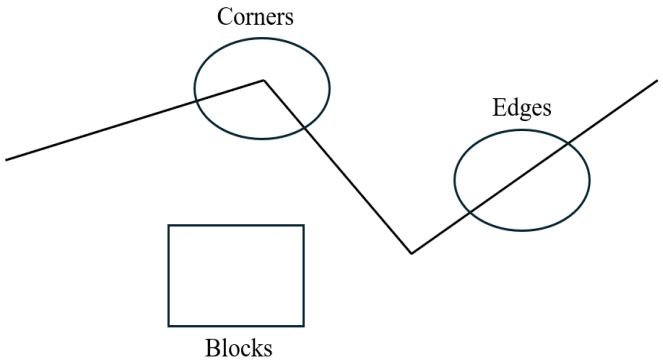


Figure 4.2: Illustration of common visual features, including corners, edges, and textured blocks, used in feature extraction techniques.

A feature point is generally described by two main components: the keypoint, which identifies the location of a distinctive region within an image, and the descriptor, which encodes the surrounding visual information for comparison and matching purposes, that is:

- Keypoint: The keypoint represents the two-dimensional location of a feature in the image. Depending on the feature extraction method, additional attributes, such as scale, orientation, or response strength, may also be associated with the keypoint to improve robustness against variations in viewpoint, rotation, and image resolution.

- Descriptor: A descriptor is a numerical representation that characterizes the local appearance around a detected keypoint. It is designed such that visually similar regions produce closely related descriptor vectors, thereby enabling reliable feature matching between different images.

There are various feature detection algorithms, including Oriented FAST and Rotated BRIEF (ORB) [88], Speeded Up Robust Features (SURF) [89], Scale-Invariant Feature Transform (SIFT) [90], and Features from Accelerated Segment Test (FAST) [91]. The Oriented FAST and Rotated BRIEF (ORB) feature extraction method is widely adopted in computer vision applications and VSLAM frameworks due to its computational efficiency and robustness [92, 93, 94]. ORB combines a modified version of the FAST detector, referred to as Oriented FAST, with the lightweight binary descriptor BRIEF. This combination significantly reduces the computational cost of feature extraction while maintaining reliable feature detection and matching performance. A popular VSLAM framework is the ORB-SLAM proposed by [95]. It leverages ORB features for robust and efficient feature detection, extraction, and matching. Figure 4.3 illustrates the typical architecture of ORB-SLAM [96]. Within the scope of the VSLAM framework investigated in this dissertation, ORB-SLAM is adopted as the baseline system for the integration and evaluation of the proposed visual data encoding methods. The framework is further analyzed and optimized in order to efficiently incorporate the developed encoding approaches while maintaining robust visual localization and mapping performance.

ORB-SLAM Operation Framework

The operation of monocular ORB-SLAM starts with map initialization, where two consecutive frames are used to construct 3D map points by extracting and matching the corresponding features. After the map is initialized successfully, the tracking extracts and matches features of the input frames to the map to localize the camera with each frame. Then, local mapping manages and optimizes the map by performing bundle adjustment of the current location in the environment. Finally, the loop closure distinguishes large loops and corrects camera trajectory drifts with the aid of a Bag of Words (BoW). The BoW is a technique that detects previously visited

areas and loops by comparing the current pose with the previously mapped ones. Successful loop closing detection enhances the accuracy of the map and the camera trajectory, emphasizing the robustness and reliability of VSLAM [97].

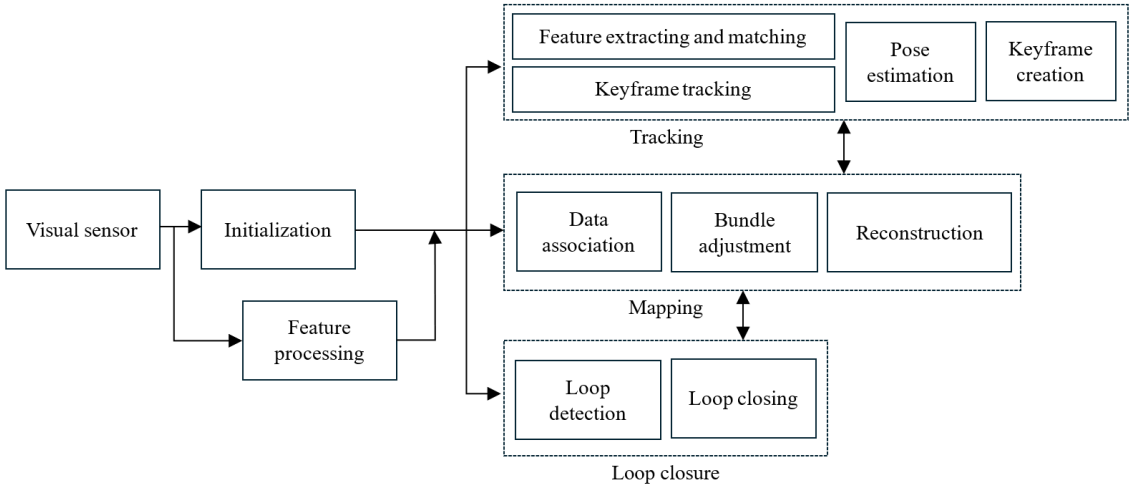


Figure 4.3: ORB-SLAM Architecture.

4.3 The Proposed End-to-End Frame Encoding Pipeline

This section demonstrates the end-to-end multi-level encoding methods tailored for the ORB-SLAM system. The proposed methods aim to optimize image compression while preserving essential visual information presented in video frames as features.

4.3.1 Multi-level Encoding Module on the Mobile Robot

The encoding algorithm is implemented on the mobile robot, which directly receives the raw image frames from the onboard camera system. The proposed approach can be integrated within both CTA and ATC frameworks, depending on the system architecture and processing requirements. In CTA, the mobile robot compresses each input frame and transmits the encoded data to the cloud. The cloud decodes the received encoded data and performs the complete ORB-SLAM processing pipeline, including tracking, mapping, loop closure detection, and global optimization. Regarding the ATC framework, the mobile device deploys both the proposed image

encoding algorithm along with the tracking module to maintain instant real-time operation. In this configuration, the tracking process is performed using raw-sized data, while the encoded visual data are transmitted to the cloud system for map reconstruction, loop-closing detection, and global optimization tasks. These framework options are selected depending on the application at hand and the computational power of the mobile device platform. Figure 4.4 illustrates the pipeline of proposed encoding method as a block diagram. The proposed method is designed to improve the efficiency and robustness of ORB-SLAM systems by introducing an optimized visual data encoding framework suitable for resource-constrained and bandwidth-limited environments. The method starts by subdividing the input frame matrix into 8×8 sub-frames followed by one level DCT to convert the frame from the spatial domain to its frequency domain. For every sub-frame, 64 frequency components are produced, one as a DC component and the other 63 as AC components. The DC component (average intensity) holds the most significant energy of the frame, while the AC components represent the variant frequency details. The DCT coefficients are then quantized using an optimized quantization matrix and scalar quantization. The optimized quantization table given in Equation (3.4) is used to improve the quality of compressed frames by preserving feature details and ensuring accurate extraction and matching of ORB features. Furthermore, in order to achieve higher compression ratios while maintaining acceptable frame quality, a multi-level encoding method is proposed to be integrated within the ORB-SLAM pipeline. The quantized coefficients are transformed into a one-dimensional array using a zigzag scanning process. Subsequently, a multi-level encoding algorithm is applied to reduce the size of the resulting array according to a user defined number of encoding levels, thereby improving compression efficiency while preserving acceptable reconstruction quality. Figure 4.5 and Figure 4.6 illustrate the proposed multi-level encoding concept with two case study examples, a single-level and two-level encoding, respectively. In both figures, K_i denotes the encoding keys (weighting coefficients) used during the coefficient reduction process. Each key is multiplied by its corresponding coefficient, and the resulting weight values are summed to generate the encoded value. In the implementation presented in this dissertation, the same set of encoding keys is reused

at every encoding level. Therefore, K_1, K_2, and K_3 in the second encoding level are identical to those used in the first level. This design simplifies the implementation while preserving the intended behavior of the proposed multi-level encoding scheme. Nevertheless, the proposed framework is sufficiently general to support different sets of encoding keys at different encoding levels.

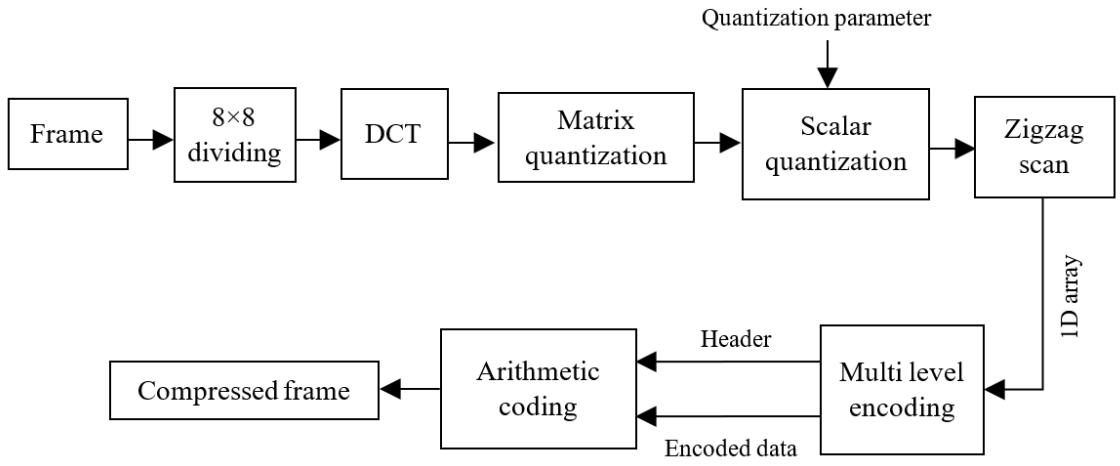


Figure 4.4: The proposed image compression pipeline.

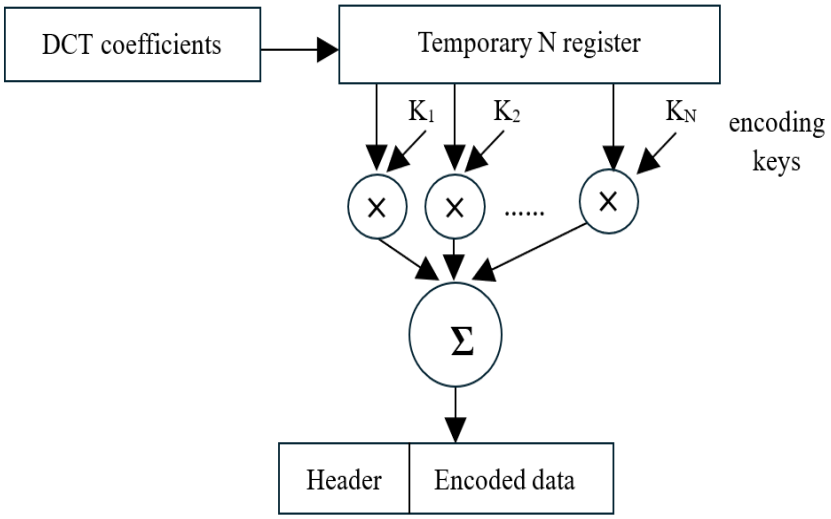


Figure 4.5: Single-level encoding implementation (the symbols (K_i) denote the encoding keys used during the coefficient reduction process).

The proposed multi-level encoding method extends the matrix reduction approach presented in Chapter 2 (Section 2.2.2) by introducing multiple successive encoding levels. At each encoding stage, the data are processed through multiplication and summation operations using a selected set of keys and their corresponding coefficient

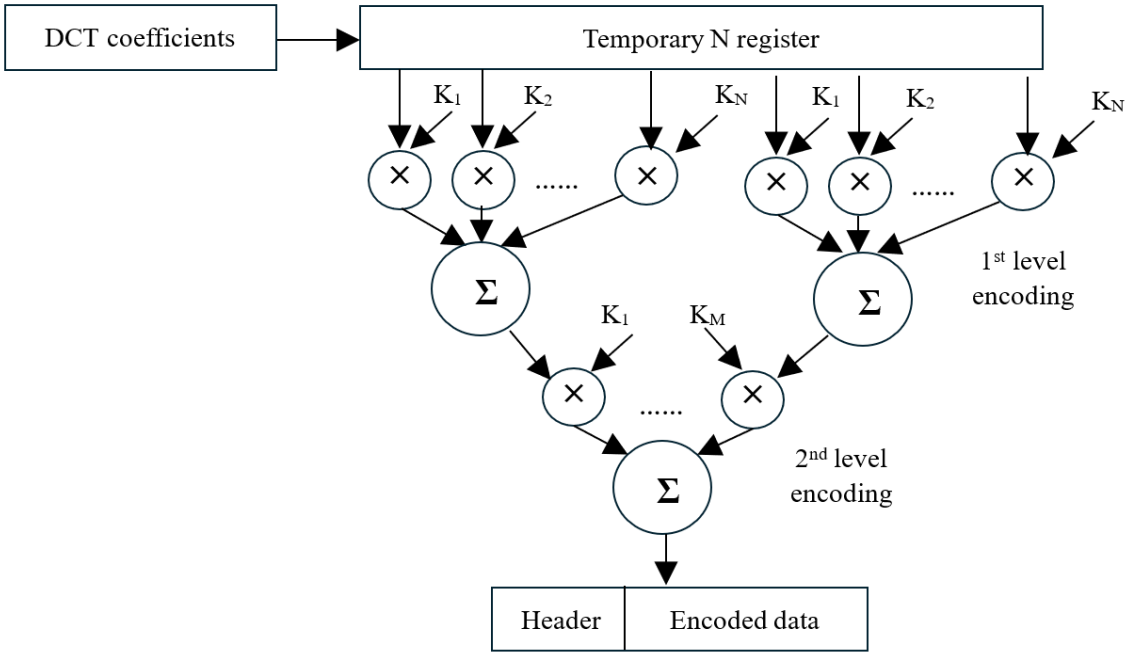


Figure 4.6: Two-level encoding implementation. The same encoding key set (K_i) is reused at both encoding levels in the proposed implementation.

values associated with the same encoding level. The generated encoded information is organized in the form of a table containing the original data probabilities as header information together with the encoded data after removing duplicate values. The resulting data are subsequently subjected to arithmetic coding for additional compression efficiency. The proposed approach significantly improves the overall compression ratio while preserving reconstruction quality, since the multi-level encoding stages operate in a lossless manner. Finally, the compressed frame data in the CTA framework, or the compressed keyframe data in the ATC framework, are transmitted to the cloud server for tracking, mapping, and loop closure operations.

4.3.2 The Decoding Module on the Edge Device

The decoding pipeline is identical to the compression process, but performed in the reverse order. The proposed decompression block diagram is shown in Figure 4.7. The decompression algorithm is deployed on the cloud server. The received data stream is first arithmetically decoded to retrieve the header and the encoded data by the proposed encoding algorithm in the compression phase. After this step, the data can be decoded using two possible methods. The first method is to use

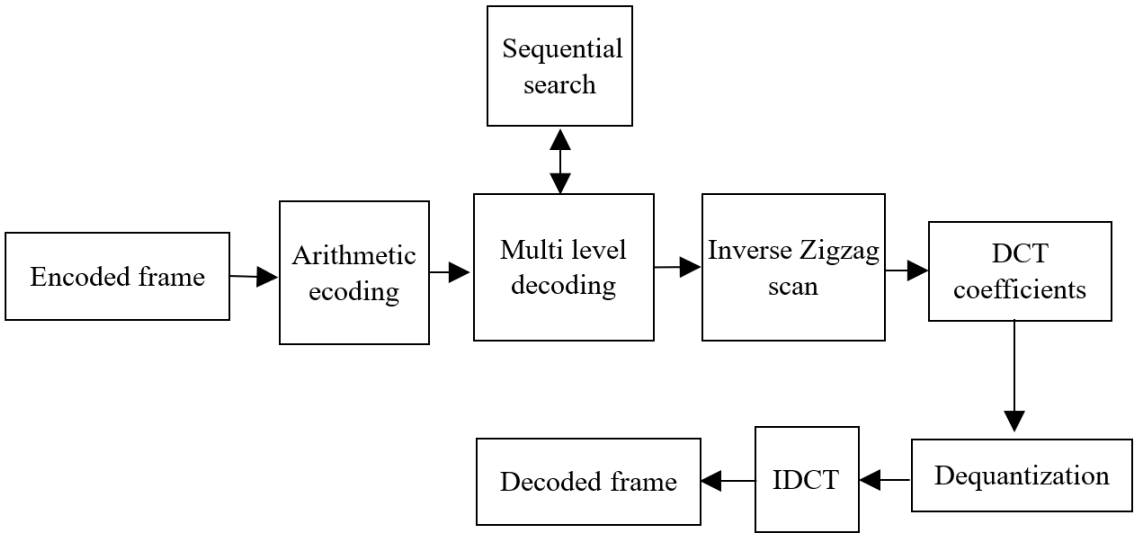


Figure 4.7: The proposed image decoding pipeline.

the header data to reconstruct the original data. In this method, the data can be decoded instantly without any extra computations. The other method involves discarding the header data during the compression phase and estimating the original data using a fast sequential search with the aid of the same encoding keys adopted during compression, as can be demonstrated in Figure 4.8. This means that we need to implement the same encoding module that was used previously in compression. This will increase the compression ratio since it only keeps the compressed data and discards the extra header data. However, the reconstruction time will be significantly increased due to the massive estimation iterations needed. This could be less problematic if we take into consideration harnessing the great cloud computing capabilities. For the scope of this work, the second method is considered to achieve a high compression ratio.

4.4 Experimental Results and Discussion

The proposed method was implemented using MATLAB R2022a, running on an Intel Core i7-12700H processor, NVIDIA GeForce RTX3070 Graphical Processing Unit (GPU) with 8 GB of memory, and 32 GB of RAM (Random Access Memory). The TUM dataset [98] is used to evaluate the implementation results and to compare the performance of the proposed methods with the traditional JPEG and JPEG2000

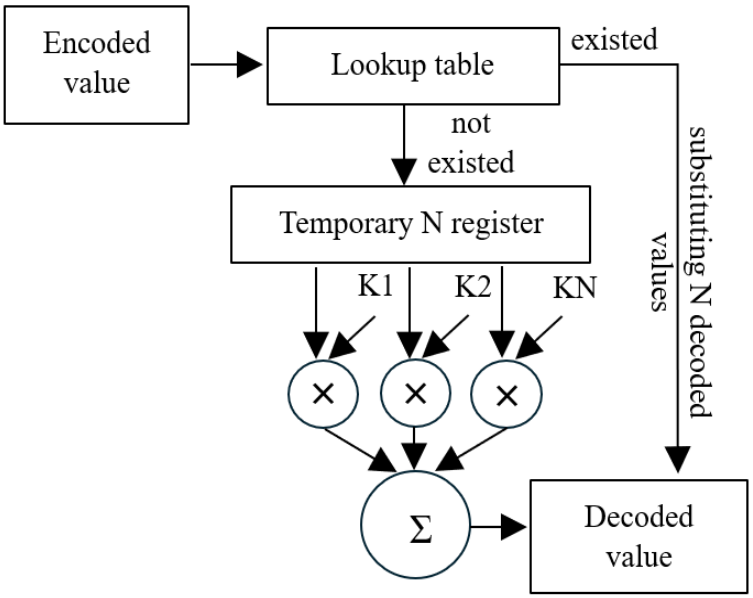


Figure 4.8: Mechanism of coefficient estimating using sequential search.

standards. It is a widely used dataset for evaluating the performance of VSLAM systems providing RGB images, depth maps, and the camera trajectory's ground truth [99]. The RGB images and ground truth of the sequence "freiburg3 long office household" are used in this work. It is an indoor dataset captured using the Asus Xtion sensor containing 2585 frames recorded at a frame rate of 30 fps. The trajectory diameter of the dataset is 5.12 m ×4.89 m×0.54 m, and its duration is 87.09 s with a frame spatial resolution of 640×480 pixels. The ending of the trajectory overlaps well with the beginning to reveal loop closure.

Firstly, the traditional ORB-SLAM is implemented when using full-sized raw visual data. The resulting estimated trajectory graph performance of this implementation is illustrated in Figure 4.9, which shows three trajectory graphs: the actual trajectory, the ORB-SLAM estimated trajectory, and the optimized trajectory. The map points are always corrected and updated according to the optimized pose trajectory graph, which is calculated from the actual trajectory and the estimated trajectory. The RMSE of the estimated trajectory is provided to evaluate the accuracy of the trajectory at each step of the simulation. The RMSE measures the absolute deviation between the estimated and ground truth trajectories. The initial robot position is defined at the coordinate origin $(0, 0, 0)$. According to the dataset used in this study,

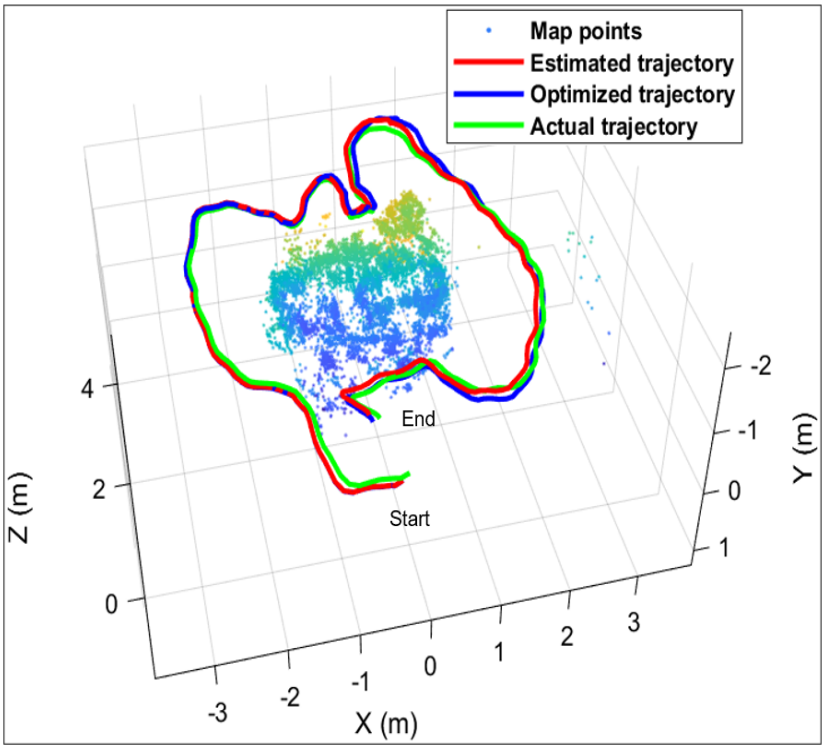


Figure 4.9: Standard ORB-SLAM trajectory estimation performance (RMSE = 0.22114).

the robot follows a clockwise trajectory, as illustrated in Figure 4.9, and similarly in the subsequent figures.

Secondly, the ORB-SLAM system is implemented by integrating the JPEG standard to encode the input data frames. The process of decoding, tracking, mapping, and loop closing is deployed as a single module emulating the cloud processing. The cloud module receives the encoded data, decodes it, and performs trajectory estimation. This cloud-based ORB-SLAM architecture will be also applied to both the JPEG2000-based system and the proposed system. In general, the purpose of using lossy image compression is to provide high compression ratios by quantizing the coefficients obtained from the discrete transformation. Therefore, additional different scalar Quantization Factors (QFs) are applied to increase the compression ratio. Furthermore, this will demonstrate the trade-off between visual data quality and trajectory estimation accuracy. Figure C.1 visualizes the trajectory estimation performance of the ORB-SLAM utilizing the JPEG standard as an end-to-end encoding and decoding framework. It illustrates the trajectory estimation among different QFs

in the range of 5 to 20 with a step value of 5. It can be deduced from Figure C.1 that JPEG compression maintained a robust ORB-SLAM function among QF value up to 15, and the loop closure in these cases has been detected accurately. However, when the quantization is increased up to 20, the trajectory drifts, and the system cannot continue mapping the remaining environment because not enough features are detected in this area. The reason behind this failure is that JPEG compression adds compression noise introduced as artifacts in the reconstructed frame, thereby negatively affecting the process of detecting and matching ORB features. Table 4.1 summarizes the ORB-SLAM performance metrics that adopted the JPEG compression. It previews the compression ratio, PSNR, SSIM, and execution time against different quantization values.

Thirdly, the ORB-SLAM is implemented by integrating the JPEG2000 standard within the ORB-SLAM architecture to encode and decode data frames. The same procedure of splitting the cloud-based ORB-SLAM tasks between the mobile device and the cloud demonstrated in JPEG-based system implementation is applied. Unlike the JPEG standard, JPEG2000 compresses the input frames according to a desired Compression Ratio (CR) parameter rather than specifying a quantization parameter. Figure C.2 shows the trajectory estimation performance of the ORB-SLAM system employing the JPEG2000 as an end-to-end encoding and decoding framework across different CRs. By examining Figure C.2, it can be observed that JPEG2000 maintained robust trajectory estimation performance up to a percentage compression ratio of 97%. However, at a higher compression ratio (CR= 98%), the system failed to detect loop closure. Table 4.2 displays the encoding and decoding performance metrics of the JPEG2000-based system.

Finally, ORB-SLAM is implemented by adopting the proposed multi-level encoding algorithm as an end-to-end visual data encoding and decoding. The trajectory estimation plots of the corresponding quantization and compression ratio conditions in both the adopted JPEG and the JPEG2000 methods are applied. The obtained simulation results of the trajectory estimation performance are presented in Figure C.3 and Figure C.4, while Table C.1 summarizes the compression performance metrics of two implementations: a single-level and a two-level encoding scheme.

Table 4.1: Encoding performance metrics of JPEG integrated into ORB-SLAM.

Frame Nr.	Original size [KB]	QF	Compressed size [KB]	PSNR [dB]	SSIM	CR [%]	Exe. time [s] Enc.	Exe. time [s] Dec.
1132	472	5	153	34.8052	0.9579	68	0.3457	0.5991
		10	144	33.2016	0.9330	69	0.2341	0.5305
		15	141	32.3478	0.9148	70	0.2279	0.5279
		20	140	29.5512	0.8842	71	0.2217	0.5071
2398	505	5	166	34.8084	0.9414	67	0.2283	0.5394
		10	164	33.2872	0.9354	68	0.2205	0.5162
		15	163	32.4000	0.9167	68	0.2195	0.5139
		20	162	30.7533	0.8874	70	0.2306	0.5212

Table 4.2: Encoding performance metrics of JPEG2000 integrated into ORB-SLAM.

Frame Nr.	Original size [KB]	CR [%]	Compressed size [KB]	PSNR [dB]	SSIM	Exe. time [s] Enc.	Exe. time [s] Dec.
1132	472	73	134	44.03	0.9823	0.110	0.440
		82	89.8	41.43	0.9721	0.136	0.412
		85	72	40.09	0.9647	0.112	0.444
		91	41	37.49	0.9483	0.195	0.532
		97	15	34.82	0.9279	0.192	0.492
		98	8.9	29.53	0.8647	0.212	0.417
2398	505	73	137.7	44.69	0.9819	0.112	0.432
		82	90	42.11	0.9708	0.121	0.403
		85	71.9	40.99	0.9460	0.115	0.515
		91	40.9	38.65	0.9485	0.109	0.636
		97	14.3	36.28	0.9303	0.181	0.636
		98	9	29.53	0.8835	0.162	0.512

The PSNR and SSIM metrics presented in Table C.1 for both methods are the same, as they achieved the same decompression quality. The reason behind the equivalent quality of the two schemes is that the multi-level encoding is lossless and does not add additional lossy compression to the encoded data. To facilitate

a direct comparison between the three evaluated compression methods, Table C.2 summarizes the compression performance of the JPEG standard, JPEG2000, and the proposed multi-level encoding method under approximately equal compression ratios. An identical compression ratio cannot be imposed for all methods because different compression control parameters are employed. Specifically, JPEG and the proposed method employ quantization that control compression as a quality factor parameter, whereas JPEG2000 directly specifies the target compression ratio. Therefore, the closest attainable compression ratios were selected to ensure a fair and meaningful compression.

As shown in Table C.2, JPEG2000 achieves the highest PSNR and SSIM among the evaluated methods. In contrast, the proposed multi-level encoding method produces the smallest compressed file size while maintaining competitive reconstruction quality. Although the proposed method requires longer encoding and decoding times than JPEG and JPEG2000 due to the additional coefficient reduction and sequential search operations, it offers a favorable trade-off between the compression efficiency and image quality, making it well suited for visual data transmission in edge-assisted VSLAM systems. It should be noted that the reported processing times are implementation-dependent. The proposed method was implemented in MATLAB for algorithmic validation, whereas JPEG2000 was evaluated using the OpenJPEG library, which is a highly optimized native C implementation. In practical edge-assisted robotic systems, optimized software implementations and hardware acceleration (e.g., FPGA or GPU-based platforms) can significantly reduce the computational overhead, while the reduced transmitted data volume can compensate for the additional processing time by lowering communication latency.

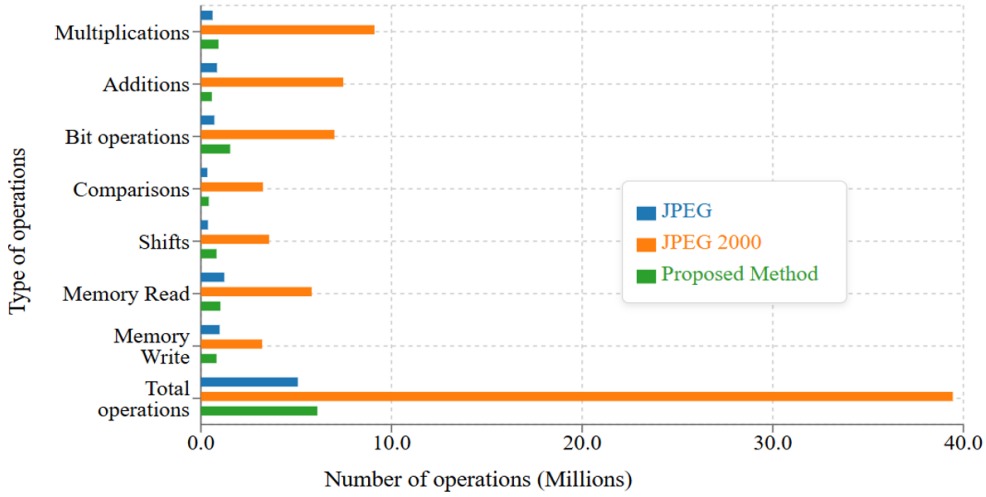
Comparing the results of the proposed system to the JPEG-based system, in all QF cases (5 to 20), the proposed method showed robust pose estimation, and loop closure was detected successfully, as clearly shown in Figure C.3. Higher values of QFs are further applied beyond the quantization values used in the JPEG-based ORB-SLAM method to evaluate the proposed method's performance under severe quantization. The proposed method showed robust trajectory estimation up to a QF value of 75. However, at higher quantization (QF=100) the proposed system failed to detect

loop closure, these cases can be shown in Figure C.4. Given the encoding metrics in Table 4.1 and Table 4.2, it can be demonstrated that the ORB-SLAM integrated with the proposed system overcomes the equivalent JPEG-based system in terms of compression ratio, PSNR, and SSIM. This clear superiority is attributed to the multi-encoding module added to the compression system. However, this addition increased the execution time, which can be seen clearly in low QF values.

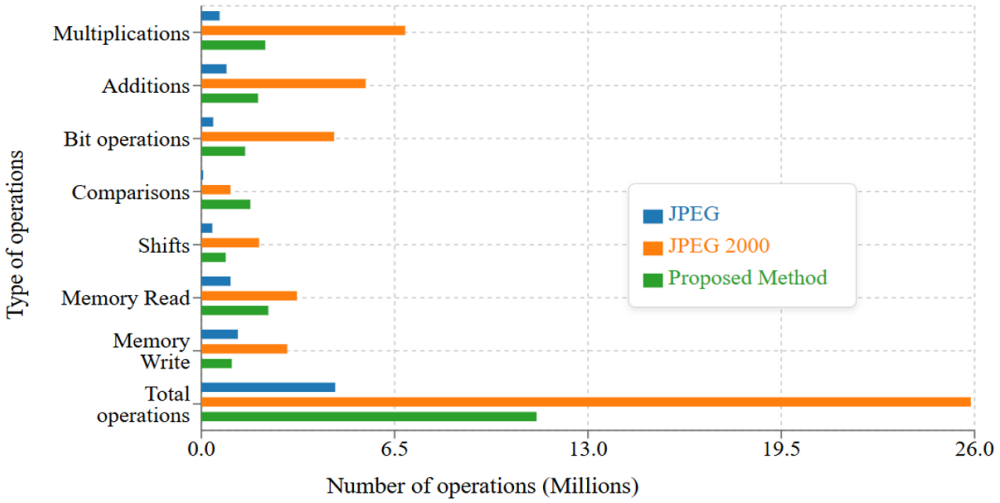
As the QF increased, more duplicated values and zeros were present in the data, yielding faster processing, smaller header sizes, smaller encoded data sizes, and higher compression ratios. Nevertheless, this higher execution time does not affect real-time operation if we consider the transmission latency, which is beyond the scope of this work; transmitting smaller data sizes could compensate for the difference in execution time.

Comparing the results of the proposed system to the JPEG2000-based system, the two systems exhibited close behaviors, with the JPEG2000-based system demonstrating slightly better performance in decoding frame quality and execution time. Nevertheless, this slight advantage in encoding metrics did not significantly enhance the overall ORB-SLAM system's performance. The proposed multi-level encoding system successfully estimated the trajectory at a compression ratio of 96%, while the JPEG2000-based system successfully estimated the trajectory at a compression ratio of up to 97%. This minor difference came at the expense of increased computational complexity. The three implemented systems adopting the JPEG, the JPEG2000, and the multi-level encoding are analyzed according to the computational complexity by calculating the total basic operations: Multiplications (Mults.), Additions (Adds.), Bit operations (Bit ops.), Comparisons (Comps.), shifts, Memory Reads (M/R), and Memory Writes (M/W) for each stage. Tables C.3, C.4, and C.5 show the computational complexity metrics for the three implemented systems processing a single input frame, while Figure 4.10 visualizes these complexity metrics.

Examining the results presented in Tables C.3, C.4, C.5, and Figure 4.10, the proposed method exhibited a clear advantage in terms of the computational complexity of both encoding and decoding modules compared to JPEG2000. This obvious ad-



a) Computational complexity of the encoding systems



b) Computational complexity comparison of the decoding systems

Figure 4.10: Computational complexity of the three implemented systems.

vantage comes with relatively equivalent trajectory estimation efficiency. On the other hand, the JPEG system shows slightly less complexity than the proposed system. However, the JPEG system showed much worse performance compared to the JPEG2000 and the proposed systems. This makes the proposed system a reasonable compromise between the three implemented systems to be used in mobile robots. Another outcome of this work is to study the impact of lossy data compression on the reliability of ORB-SLAM systems. The encoding performance results presented in Tables 4.1, 4.2, and C.1 demonstrated that ORB-SLAM is less robust when PSNR values go below 30 and SSIM values below 0.87.

4.5 Thesis III.

I developed and implemented a novel multi-level encoding method for visual data compression specifically tailored for robust monocular ORB-SLAM systems operating in cloud-edge computing environments. This comprehensive approach addresses the critical challenge of bandwidth optimization in visual SLAM applications while maintaining trajectory estimation accuracy and feature detection quality..

4.5.1 Sub-thesis III.1

I have designed and validated two distinct multi-level encoding schemes: a single-level implementation and a two-level implementation, both integrating seamlessly with the ORB-SLAM architecture.

4.5.2 Sub-thesis III.2

I have demonstrated that the proposed multi-level encoding method significantly outperforms traditional JPEG and JPEG2000 compression in terms of robustness under varying quantization factors, making it more suitable for resource-constrained mobile devices.

Related Publications: [P3], [P6], [P7].

4.6 Chapter Findings

This chapter presented a novel multi-level visual data encoding framework tailored for cloud- and edge-assisted ORB-SLAM systems operating under bandwidth and computational constraints. The proposed method is successfully integrated within both Compress-Then-Analyze and Analyze-Then-Compress architectures and evaluated using the ORB-SLAM framework on the TUM RGB-D dataset.

Compared with the conventional JPEG-based ORB-SLAM system, the proposed approach provided superior decoded frame quality and more reliable feature pre-

servation at higher quantization levels, resulting in improved trajectory estimation accuracy and successful loop closure detection under severe compression conditions. Furthermore, the proposed system demonstrated performance comparable to the more computationally demanding JPEG2000-based ORB-SLAM system, while exhibiting considerably lower computational complexity. The complexity analysis confirmed that the proposed method offers a practical compromise between compression efficiency, reconstruction quality, and computational cost, making it suitable for resource-constrained mobile robotic platforms. The results also revealed that ORB-SLAM performance becomes increasingly sensitive when the reconstructed frame quality falls below approximately 30 dB PSNR and 0.87 SSIM. In addition, increasing quantization levels reduced execution time due to the larger number of duplicated coefficients and zero values generated during encoding. Although the proposed algorithm introduces a noticeable processing delay in decoding operations, it can still be efficiently deployed on cloud server platforms equipped with high-performance computational resources and powerful hardware capabilities.

Chapter 5

Visual Data Encoding in Multi-Robot Visual SLAM Systems

5.1 Literature Background and Introduction

Visual simultaneous localization and mapping has gained significant attention in robotics and autonomous navigation applications due to the affordability of camera sensors and their strong capability for spatial environmental perception, despite the continued dominance of LiDAR-based SLAM systems in these domains. Although a large number of VSLAM methods and frameworks have been proposed, relatively limited research has focused on multi-agent scenarios. Such environments may involve teams of autonomous ground robots, aerial drones, or multiple users operating augmented reality devices within a shared environment. In these situations, agents are required not only to operate independently, but also to maintain awareness of the motion and mapping information generated by other agents in the system. For instance, an aerial drone operating within a swarm must estimate both its own pose for navigation and the poses of neighboring drones to support collision avoidance and cooperative path planning. Furthermore, the three-dimensional maps generated by different agents should be efficiently fused in order to accelerate exploration and improve the consistency of the reconstructed environment. [100]. A multi-robot system consists of a group of robots (agents) equipped with perception sensors. It extends the capabilities of a single robot by merging the measurements of multiple team members, providing each robot with information beyond the range of its single

sensor. This allows for more efficient use of resources and enables tasks that cannot be accomplished by a single robot system. In addition, the use of a multi-robot system allows tasks to be performed in parallel with a certain degree of redundancy, which improves the efficiency and robustness of the system. The information collected from mobile robots can be used for a wide range of applications, such as exploration strategies, path planning, building shared maps, and other higher-level decisions. Let us consider a scenario where a multi-robot system is deployed to map a large unknown environment. Tasks can be assigned to all team members who can work together to create a global map, thereby reducing the overall execution time [101]. However, the benefits of multi-robot systems come at the expense of increased computation and communication load between robots.

Efficient visual data transmission has become a critical requirement in cloud-based VSLAM systems, particularly in bandwidth-constrained robotic and edge-assisted environments. A large amount of research effort has been devoted to collaborative multi-robot systems in recent years. However, studies investigating the impact of visual data size and compression quality on the performance of collaborative VSLAM systems remain relatively limited. A JPEG-based compression scheme is proposed by [102] to compress the point cloud data for the sharing environment. The algorithm used is inspired by JPEG and compresses the sharing environment as a panorama. The studies [103, 104] addressed the challenge of efficient data sharing in multi-robot systems by comparing various Free and Open Source Software (FOSS) lossless compression techniques for optimizing communication bandwidth. They focused specifically on compressing occupancy grid maps in cooperative multi-robot SLAM scenarios. The studies concluded that general purpose compression techniques are viable solutions for reducing bandwidth requirements in multi-robot systems, with different algorithms offering various trade-offs between compression efficiency and computational cost. Similar work [105] investigates the problem of communication for multi-robot SLAM systems. It employed lossless encoding by combining an intelligent Huffman method and Run-Length Encoding (RLE) for full map compression before transmission. In addition, they incorporate lightweight Convolutional Neural Networks (CNN) for feature retrieval. A lossless compression and encoding method

for keyframe feature descriptors in the centralized ORB-SLAM3 architecture is introduced in [106]. Similarly, a quantized redundant keyframe deletion algorithm is proposed in [107] to reduce the consumed time caused by large data volumes and improve real-time performance in multi-robot systems. Along the same direction, a collaborative visual SLAM system for multiple robots (CORB-SLAM) architecture is presented by extending the single robot ORB-SLAM2 architecture to support multi-robot systems. A memory management module is added to the mobile robot architecture to organize the local mapping and communication tasks with the server. The server fuses the local maps of the mobile robots into a shared global map. In [77], Edge-SLAM architecture is implemented by leveraging edge computing to address resource constraints in mobile devices. To address communication latency, three parallel connections are implemented for features and local geometry, candidate keyframes, and updating the local map at the edge.

Efficient communication is critical to ensure low latency and reliable performance. Many studies treated the compression and encoding of visual data such as features and map points using lossless techniques. In contrast, lossy compression can provide an efficient solution in compression of visual data for the V-SLAM system. However, relatively few studies have investigated lossy compression techniques for multi-robot V-SLAM systems. Building upon the visual data encoding method introduced in Chapter 4, lossy compression techniques provide a practical solution for reducing overhead while maintaining acceptable mapping performance. A JPEG-developed algorithm is proposed to compress visual data between the mobile robot and the cloud. However, this work was limited to a single-robot architecture. The contributions of this chapter are summarized as follows:

- An efficient centralized edge-based multi-robot architecture is presented by extending the single-robot ORB-SLAM architecture. The main V-SLAM tasks are distributed between the mobile robot and the edge device.
- An efficient communication module is developed to enhance encoding performance without adversely affecting VSLAM accuracy.
- Through extensive simulation results, the work provides insight into the effects

of lossy compression of visual data on the multi-robot V-SLAM platforms in terms of robustness and trajectory accuracy.

5.2 Methodology

Although Visual SLAM has made progress in recent years, it is difficult for a single mobile robot to complete complex tasks in large and complex scenarios. Multi-robot systems improve the efficiency of individual robots through communication, collaboration, and information exchange, such as complex tasks, robustness, flexibility, and fault tolerance of task execution. Due to the limitations of single robot SLAM in tasks and the strong flexibility and adaptability of multi-robot systems, collaborative multi-robot SLAM has attracted more and more attention. Edge-assisted computing represents an effective solution for the problems of limited computing resources, communication, and storage capabilities of terminal devices. However, the massive communication loads and limited bandwidth between the terminal devices and the edge contribute significantly to degrading the performance of SLAM systems [108]. To investigate these challenges, a collaborative multi-robot ORB-SLAM framework consisting of two mobile robots was implemented as a case study. In the proposed architecture, each robot performs real-time tracking and local mapping independently, while an edge device is responsible for global map fusion, loop closure detection, and trajectory optimization. The implemented framework enables the exchange of compressed visual information between robots and the edge server, allowing efficient collaborative mapping while reducing communication overhead and onboard computational requirements. This setup also provides a practical platform for evaluating the impact of the proposed visual data encoding methods on transmission efficiency, localization accuracy, and global map consistency in multi-robot environments.

5.2.1 Proposed Multi-Robot V-SLAM System Architecture

A multi-robot visual SLAM-based centralized edge with a global map reconstruction system is proposed in analyze then compress paradigm. The ORB-SLAM architecture is distributed between the mobile robots and the edge. Each robot includes

tracking, local mapping, and communication modules to process and transmit the data to the edge, which in turn, performs the robot's map optimization, fusing the global map from the maps of both robots and loop closure. Figure 5.1 illustrates the implemented multi-robot ORB-SLAM architecture, which employs two robots as a case study, and can be extended to include more.

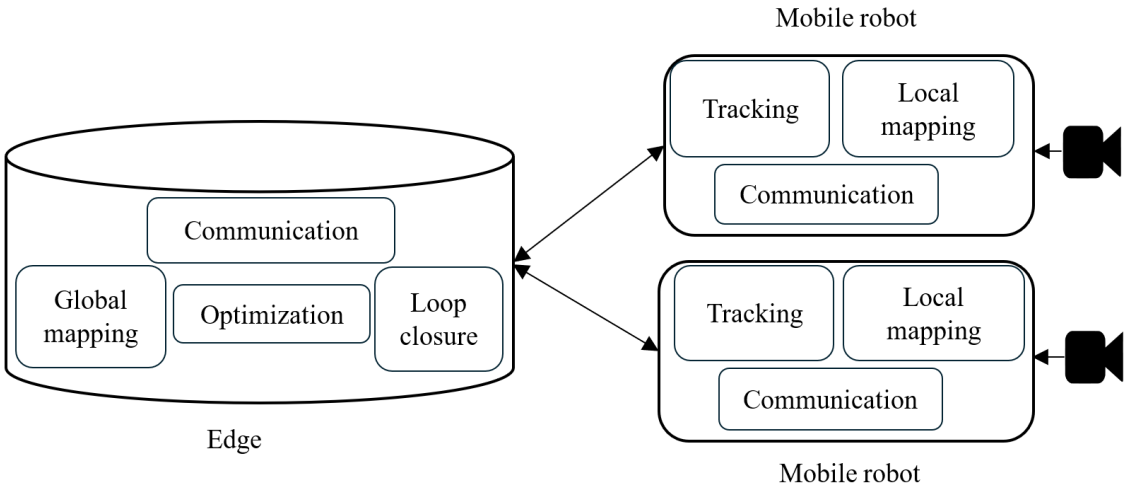


Figure 5.1: Proposed multi-robot ORB-SLAM architecture.

5.2.2 Proposed Communication Module

The proposed communication module consists of two separate parts: encoder and decoder. The encoder is designed to be deployed on the mobile robot to encode the candidate keyframes from the tracking module. To design such an encoder, the key requirements are to be as simple as possible, have low latency to maintain real-time performance, and minimal resource-consumption. These factors realize efficient and low-power mobile robots. Figure 5.2 illustrates the basic block diagram of the proposed communication module.

A single level encoding method, introduced in Chapter 4 (Section 4.3.1), is adopted and further developed to improve compression efficiency and accelerate the decoding process. Three scenarios are proposed for reconstructing the minimized DCT coefficients, depending on where the dictionary and sequential search stages are enabled or omitted within the encoding and decoding pipelines, respectively. The communication module architecture is illustrated in Figure 5.2, where the mentioned stages

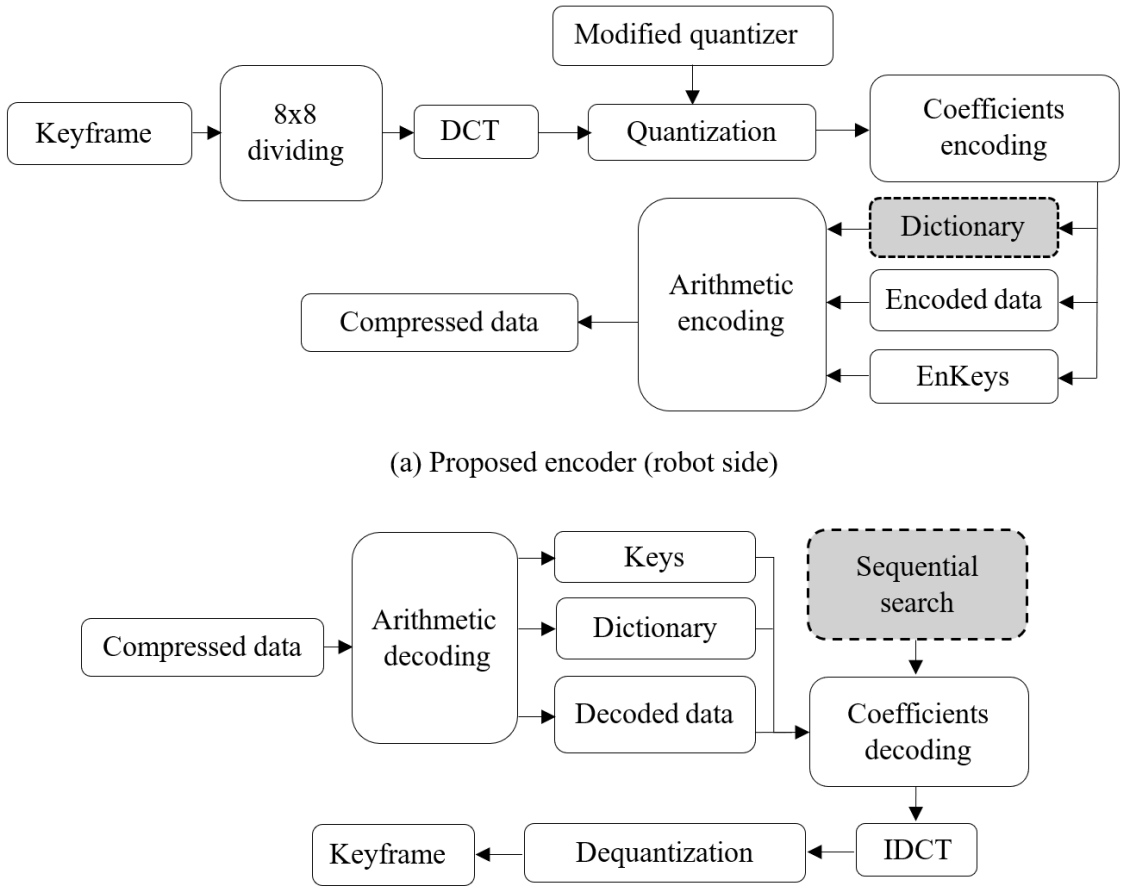


Figure 5.2: Proposed communication pipeline.

are highlighted as shaded blocks. These three proposed scenarios are described as follows:

Sequential Search Method

The sequential search method uses the dictionary attached in the compressed frame's header, as well as encoding keys, to estimate the original DCT coefficients. To speed up the search process and avoid redundant blind searches, a fast sequential algorithm is implemented. This algorithm generates a cache table to store the estimated decoded values for each matched result. Consequently, subsequent searches for the same encoded value can be bypassed by substituting the decoded data directly from the cache table and eliminating the need for further search. Conversely, if the calculated result does not match the encoded one, the search continues to estimate a

new output using other probabilities within the unique table. This mechanism is illustrated in Figure 5.3 and described in Algorithm 1. Here, ED_i denotes the ith encoded value, DD_j denotes the corresponding decoded data, and K_N denotes the Nth encoding key employed by the sequential search algorithm to reconstruct the original coefficient values.

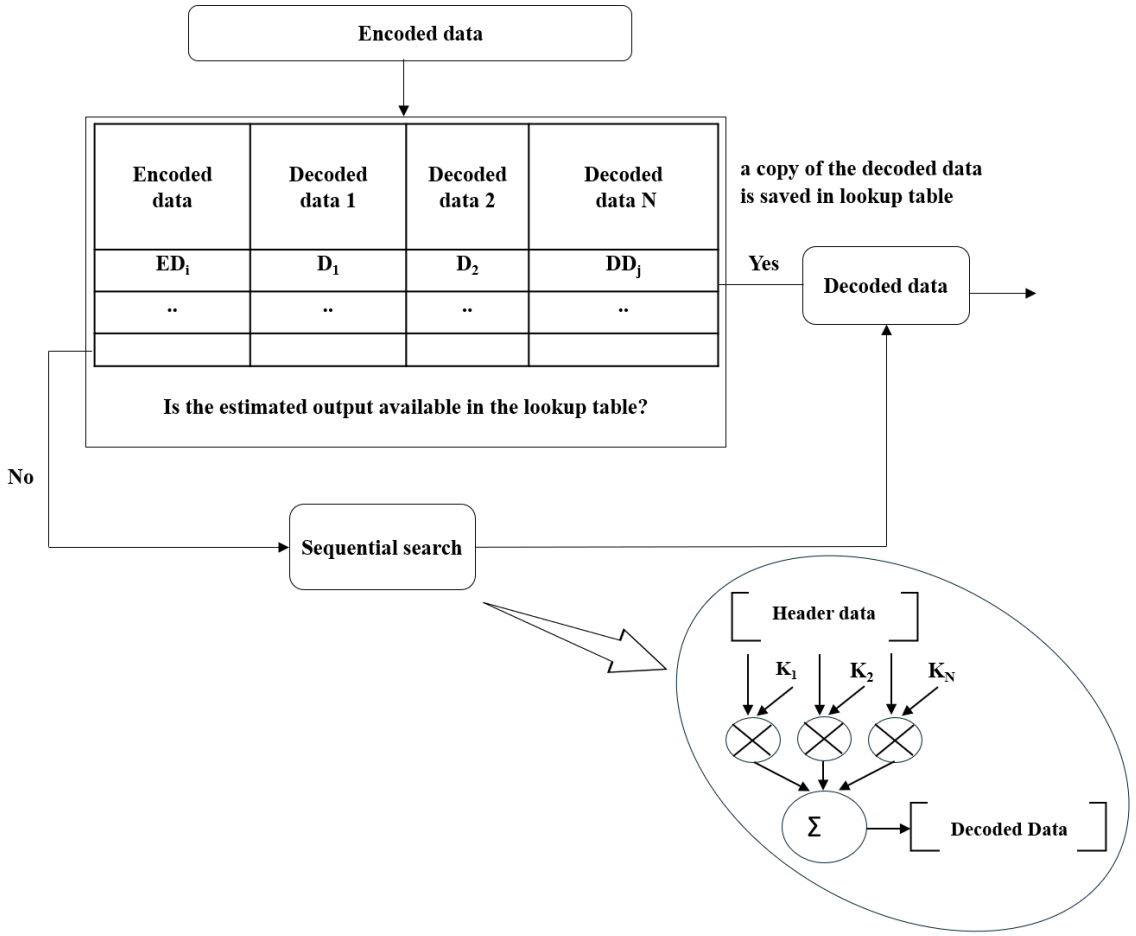


Figure 5.3: Proposed sequential search mechanism for coefficient decoding.

Direct calculation method

It decodes a list of encoded values by directly reversing the encoding process. Each encoded value represents a combination of three original coefficients packed using a weighted sum with encoding keys. This function recovers the original three coefficients from each value using basic arithmetic (division and modulo operations), without needing any lookup tables. Note that in this case, the encoding keys should be the powers of a base number that is larger than the maximum value in coefficient

Input: encodedData, uniqueTable, keys
Output: decodedData
Initialize $L \leftarrow 1$;
Initialize decoded data as an empty list;
Initialize cacheArray with one row: $[0, 0, 0, 0]$;

for *each value in encoded data* **do**
    if *value exists in the first column of the cacheArray* **then**
        Retrieve the corresponding row;
        Append values from columns 2, 3, and 4 of that row to decoded data;
        Increment L by 3;
    end
    else
        $cacheArray \leftarrow estimateData(keys, uniqueTable, value)$;
        Append $cacheArray[0], cacheArray[1], andcacheArray[2]$ to decoded data;
        Increment L by 3;
        Append the following row to cacheArray;
        $[value,\ cacheArray[0],\ cacheArray[1],\ cacheArray[2]]$;
    end
end
return decoded data;

Algorithm 1: Sequential Searching Scenario

data to create unique mappings for all possible triplets. Algorithm 2 declares this concept.

Input: encodedData
Output: decodedData
Initialize decoded data with size equal to the length of encoded data $\times$ 3;
Set keys to $[1,\ base_value^1,\ base_value^2]$;

for *each value in encoded data* **do**
    Round the value to the nearest integer;
    $coef3 \leftarrow \lfloor value/key3 \rfloor$;
    $remainder \leftarrow value - coef3 \times key3$;
    $coef2 \leftarrow \lfloor remainder/key2 \rfloor$;
    $coef1 \leftarrow remainder - coef2 \times key2$;
    Append $coef1$, $coef2$, and $coef3$ to decoded data;
end

return decoded data;

Algorithm 2: Direct Calculation Scenario

Table method:

The table method involves constructing a large, predefined table on the edge side to map each value of encoded data back to its original three coefficients data. Each row of the table contains an encoded value in the first column and its three corresponding original coefficients in the next three columns. If a match is found, the corresponding coefficients are added to the output vector. The table-based procedure is presented in Algorithm 3.

Input: encodedData, table
Output: decodedData
Initialize decoded data with size equal to the length of encoded data $\times$ 3;
for *each value in encoded data* **do**
    Search for a matching value in the first column of the table;
    if *match found* **then**
        Extract the three corresponding coefficients from the table;
        Append them to decoded data;
    end
    else
        Append $[0, 0, 0]$ to decoded data;
    end
end
return decoded data;

Algorithm 3: Table Scenario

5.3 Results and Discussion

To evaluate the performance of the proposed architecture and demonstrate the effectiveness of the distributed architecture and the communication module of the multi-robot system, the proposed system was implemented and simulated using MATLAB R2025b. The hardware environment used in this work is a PC equipped with an Intel Core-i7 12700H processor, a 32 GB of Random Access Memory (RAM), and an NVidia GeForce RTX3070 Graphical Processing Unit (GPU) with 8 GB of memory. The dataset used for the simulation is the TUM RGB-D dataset "freiburg3 long office household", which is mentioned and explained in Section 4.4. The dataset includes 2585 frames partitioned between the two robots with sufficient overlap to ensure proper global mapping and loop closure detection. The proposed system is

evaluated in two stages. Firstly, the multi-robot architecture is simulated using the original-size frames to examine the designed system performance and global map fusion. Secondly, the system is simulated using different scalar Quantization (Q) values to evaluate the performance of the proposed communication module. The RMSE is used to evaluate the accuracy of the trajectories at each step of the simulation. The RMSE measures the absolute deviation between the estimated and the corresponding ground truth trajectories.

In the first stage, the proposed system is evaluated using the original-size frames. In this configuration, robot-1 and robot-2 capture raw visual frames, initialize their maps, and track candidate keyframes before transmitting them to the central edge server. On the edge side, the received visual data from both robots are processed to optimize the local maps and fuse them into a unified global map. Figures D.1 and D.2 present the estimated trajectories of robot-1 and robot-2, respectively, compared with the corresponding ground truth trajectories. Furthermore, the combined global map generated at the edge side, illustrating the overall fused trajectory, is shown in Figure D.3. The obtained results demonstrate that the proposed architecture is capable of performing reliable visual SLAM operation for both robots while maintaining accurate trajectory estimation in terms of RMSE. In addition, the edge server successfully performed global map fusion using the estimated poses from both robots, resulting in a convergent trajectory with acceptable location accuracy.

In the second stage, various quantization values ranging from low to severe compression levels are applied to the encoding module in order to evaluate the system performance under progressively degraded visual quality. The objective is to reduce the transmitted data size and bandwidth utilization while maintaining reliable VSLAM operation. This stage allows assessment of the robustness limits of the proposed system compared to the existing compression standards (JPEG and JPEG2000) and determines the extent to which data compression can be applied while still preserving acceptable localization and mapping performance. Table 5.1 presents the encoding performance of the proposed system. The evaluation was performed using quantization values ranging from 20 to 100 with a step value of 20.

Table 5.1: Communication performance key metrics for the proposed multi-robot ORB-SLAM architecture.

Quantization factors (Q)						
Details	**1**	**20**	**40**	**60**	**80**	**100**
No. of keyframes						
Robot-1	72	66	99	72	109	79
Robot-2	67	68	72	70	77	74
Edge	139	134	171	142	186	153
No. of map points						
Robot-1	5674	4105	5221	3795	4677	3438
Robot-2	6068	5662	5442	4973	4643	4268
Edge	11742	9767	10663	8768	9320	7706
RMSE [m]						
Robot-1	0.0658	0.0611	0.0756	0.0712	0.1018	0.1472
Robot-2	0.0595	0.0781	0.1370	0.1143	0.1765	0.1276
Edge	0.0635	0.0767	0.1066	0.1492	0.1323	0.1357
Memory size [KB]						
Robot-1	132.98	96.21	122.37	88.95	109.62	80.58
Robot-2	142.22	131.81	127.55	116.55	107.98	100.03
Edge	275.20	228.02	249.91	205.50	217.59	180.61
Memory reduction [%]						
Robot-1	n/a	28	8	33	18	39
Robot-2	n/a	7	10	18	24	30
Edge	n/a	17	9	25	21	34

It can be observed from Table 5.1 that the proposed system maintained robust operation under different quantization settings. This behavior is further illustrated by the combined map trajectories shown in Figure D.4 at $Q = 100$, where the system still preserves reliable global map reconstruction and trajectory consistency. However, at higher quantization values beyond $Q = 100$ (e.g. $Q = 120$), noticeable errors begin to appear in the combined global map, as illustrated in Figure D.5. Beyond this threshold, the system becomes unable to reconstruct a reliable global map at the edge side. This degradation is mainly attributed to the severe compression artifacts

introduced at high quantization levels, which reduce the accuracy of ORB features matching across different robot viewpoints. Consequently, the accumulated pose estimation error increases, leading to geometric distortion and reduced reliability in global map alignment.

The proposed system is compared with a corresponding multi-robot architecture employing JPEG as the reference compression module. Table 5.2 presents the communication performance metrics obtained using the JPEG-based framework. The results indicate that the JPEG-based communication model maintains stable operation up to $Q = 60$. However, beyond this quantization level, the system fails to successfully fuse the global map due to the increasing trajectory estimation errors introduced by severe JPEG compression artifacts. The degradation in reconstructed frame quality negatively affects feature matching and pose estimation consistency between the two robots, leading to accumulated localization errors and unreliable global map alignment. In contrast to both the proposed and JPEG-based systems, the JPEG2000 standard is evaluated using predefined target compression ratio settings, since compression ratio control represents the primary configuration parameter of the JPEG2000 framework. Table 5.3 presents the performance of the JPEG2000-based multi-robot system.

The results presented in Tables 5.1, 5.2, and 5.3 demonstrate the impact of different compression methods on the communication performance and mapping accuracy of the multi-robot ORB-SLAM system. Compressing the transmitted keyframes significantly reduces the communication memory requirements at both the robot and edge levels, thereby decreasing bandwidth utilization and supporting more efficient real-time operation in resource-constrained environments. The memory reduction percentage is calculated relative to the original communication memory size obtained at $Q = 1$ using Equation (5.1):

$$\text{Memory Reduction } (\%) = \frac{M_{\text{original}} - M_{\text{compressed}}}{M_{\text{original}}} \times 100 \tag{5.1}$$

where M_{original} represents the memory size at the original quality level and $M_{\text{compressed}}$ denotes the corresponding compressed memory size.

Table 5.2: Communication performance key metrics for a corresponding JPEG-based architecture.

Quantization factors (Q)			
Details	**20**	**40**	**60**
No. of keyframes			
Robot-1	89	112	115
Robot-2	103	82	108
Edge	192	194	223
No. of map points			
Robot-1	3386	3756	4054
Robot-2	3669	2745	3186
Edge	7055	6501	7240
RMSE [m]			
Robot-1	0.0663	0.1079	0.1056
Robot-2	0.1197	0.1056	0.2741
Edge	0.1144	0.1852	0.1114
Memory size [KB]			
Robot-1	79.359	88.031	95.016
Robot-2	85.992	64.336	74.672
Edge	165.35	152.37	169.96
Memory reduction [%]			
Robot-1	40	34	29
Robot-2	40	55	74
Edge	40	45	38

A detailed comparison of Tables 5.1 and 5.2 shows that the JPEG-based architecture achieves the highest memory reduction, reaching up to 55% for Robot-2. However, this improvement is accompanied by a noticeable degradation in localization performance at higher compression levels. For example, the RMSE of Robot-2 increases to 0.2741 m at $Q = 60$, indicating that the compression artifacts introduced by JPEG adversely affect feature extraction, feature matching, and pose estimation accur-

Table 5.3: Communication performance key metrics for a corresponding JPEG2000-based multi-robot architecture.

Details	**Percentage compression ratio (CR%)**				
	80	**85**	**90**	**95**	**99**
No. of keyframes					
Robot-1	66	66	67	66	66
Robot-2	73	76	73	84	84
Edge	139	142	140	150	150
No. of map points					
Robot-1	6306	6355	6210	5903	5903
Robot-2	5303	5597	7313	6277	6277
Edge	11609	11952	13523	12180	12180
RMSE [m]					
Robot-1	0.0414	0.0701	0.0541	0.1018	0.1798
Robot-2	0.05421	0.1121	0.0967	0.0862	0.1762
Edge	0.05424	0.09778	0.1274	0.1408	0.1874
Memory size [KB]					
Robot-1	147.29	148.95	145.55	138.35	138.35
Robot-2	124.29	131.18	173.09	147.12	147.12
Edge	272.09	280.12	318.63	285.47	285.47
Memory reduction [%]					
Robot-1	-11	-12	-9	-4	-4
Robot-2	12	8	-21	-3	-3
Edge	1	-1	-15	-3	-3

acy. Furthermore, the JPEG-based architecture produces fewer reconstructed map points than the proposed method, suggesting a greater loss of geometric information required for reliable map generation and global map fusion.

In contrast, the proposed architecture provides a more balanced trade-off between communication efficiency and V-SLAM performance. Although its maximum memory reduction is generally lower than that of the JPEG-based system, reaching 39%, 30%, and 34% for Robot-1, Robot-2, and the edge device, respectively, it maintains robust

localization performance over a wider range of compression settings. The proposed method consistently preserves a larger number of map points while maintaining lower RMSE values, demonstrating greater robustness against compression-induced image degradation. Consequently, reliable trajectory estimation and global map reconstruction can still be achieved even at high quantization values.

Table 5.3 provides an additional comparison with a corresponding JPEG2000-based architecture. The results indicate that JPEG2000 preserves image structures and visual features more effectively than conventional JPEG, resulting in a larger number of reconstructed map points and competitive localization accuracy. However, this improvement is achieved at the expense of substantially higher communication memory requirements. In several cases, the memory consumption of the JPEG2000-based architecture approaches or exceeds the reference communication size, resulting in limited memory savings despite the high target compression ratios. Therefore, while JPEG2000 offers strong reconstruction quality, its communication efficiency is less attractive for bandwidth-constrained edge-assisted robotic systems.

For additional analysis, Table D.1 summarizes the compression and decompression performance metrics of a sample keyframe encoded and decoded by the proposed, the JPEG-based, and the JPEG2000-based ORB-SLAM systems, respectively. It presents Compression Ratio, Peak Signal-to-Noise Ratio, Structural Similarity Index Measure, and execution time against different quantization values (for the proposed and JPEG-based systems) and CR values (for JPEG-2000 based system). It should be noted that the searching method has a distinct encoding time because, in addition to the compression process, it generates and stores the unique probability information required for sequential searching during the decoding stage. In contrast, the direct calculation and table methods share the same encoding time as both employ an identical encoding procedure and do not include any additional reconstruction information in the compressed bitstream. Instead, the original coefficients are reconstructed during decoding either through direct mathematical calculations using predefined encoding keys or by searching a precomputed lookup table. Consequently, the direct calculation and table methods have identical encoding times, whereas their decoding times differ according to the adopted coefficient reconstruction strategy.

As evident from the data given in Table D.1, the proposed system demonstrated outstanding performance compared to the JPEG-based system in terms of compression ratios and reconstructed visual data quality. However, The JPEG-based system showed a better execution time in the compression stage and an equivalent processing time in the decompression. This is attributed to the additional computational tasks by the proposed DCT coefficients minimization module. Nevertheless, if considering transmitting data under higher compression ratio, the overall execution time will be significantly lower. In contrast, examining the data regarding the proposed and the JPEG-2000 systems, both show similar and equivalent performance with remarkable better quality of reconstructed frame in the JPEG2000-based system in corresponding compression ratios. The JPEG2000 has slight advantages in terms of compression ratio compared to proposed-searching method, while the proposed system shows better compression ratio in the direct calculation and table methods. This improvement is attributed to discarding the unique table and not attaching it in the frame header as these methods can estimate original data using the encoding keys only. Although lookup-table decoding is generally expected to provide fast access through direct indexing, the implemented table method exhibited a longer execution time in the MATLAB environment. The lookup table is constructed by exhaustively storing all possible combinations of quantized DCT coefficient triplets together with their corresponding encoded values in a four-column mapping table, resulting in a large memory-intensive data structure. Consequently, repeated memory accesses during decoding introduce indexing overhead and reduce cache efficiency. In contrast, the direct calculation method reconstructs the coefficients analytically using only integer arithmetic operations, including integer division, multiplication, subtraction, and modulo operations, without accessing a lookup table. Therefore, in the current MATLAB implementation, the direct calculation method achieved a lower decoding time than the table-based approach and demonstrated a decoding performance comparable to that of the JPEG2000-based system. It should be noted that this observation is implementation-dependent. The reported decoding times correspond to the MATLAB software implementation used for algorithm val-

idation. More efficient software implementations or hardware-oriented realizations may exhibit different performance characteristics.

Furthermore, Table D.2 presents the computational complexity analysis of the proposed method, including the encoding process and the searching, direct-calculation, and table-based decoding implementations. The analysis considers multiplications, additions, bit operations, comparisons, shifts, memory reads, and memory writes. For comparison, the corresponding complexity results for the JPEG- and JPEG2000-based systems were previously presented in Tables C.3 and C.4 of Chapter 4, respectively. The overall comparison given in Figure 5.4 shows that the encoding complexity of the proposed method is close to that of JPEG and considerably lower than that of JPEG2000. For decoding, the direct-calculation implementation has the lowest complexity among the proposed methods, followed by the searching implementation, and both require fewer operations than JPEG2000. In contrast, the table-based implementation is considerably more demanding, primarily because of the large number of bit operations, comparisons, and memory reads associated with the table-searching process. Although JPEG has the lowest overall computational complexity, its performance deteriorates under high-compression conditions, as demonstrated by the experimental results. The proposed method, particularly with direct-calculation decoding, therefore provides a suitable compromise between computational complexity, compression efficiency, and the preservation of visual information required for VSLAM.

In summary and as the results demonstrate, the proposed system overcomes the JPEG-based system in both ORB-VSLAM approaches, while demonstrating performance equivalent to the more recent version JPEG2000 standard. This makes the proposed system utilizing the searching and direct calculation decoding methods a reasonable compromise between the three implemented systems to be used in resource-constrained mobile robots.

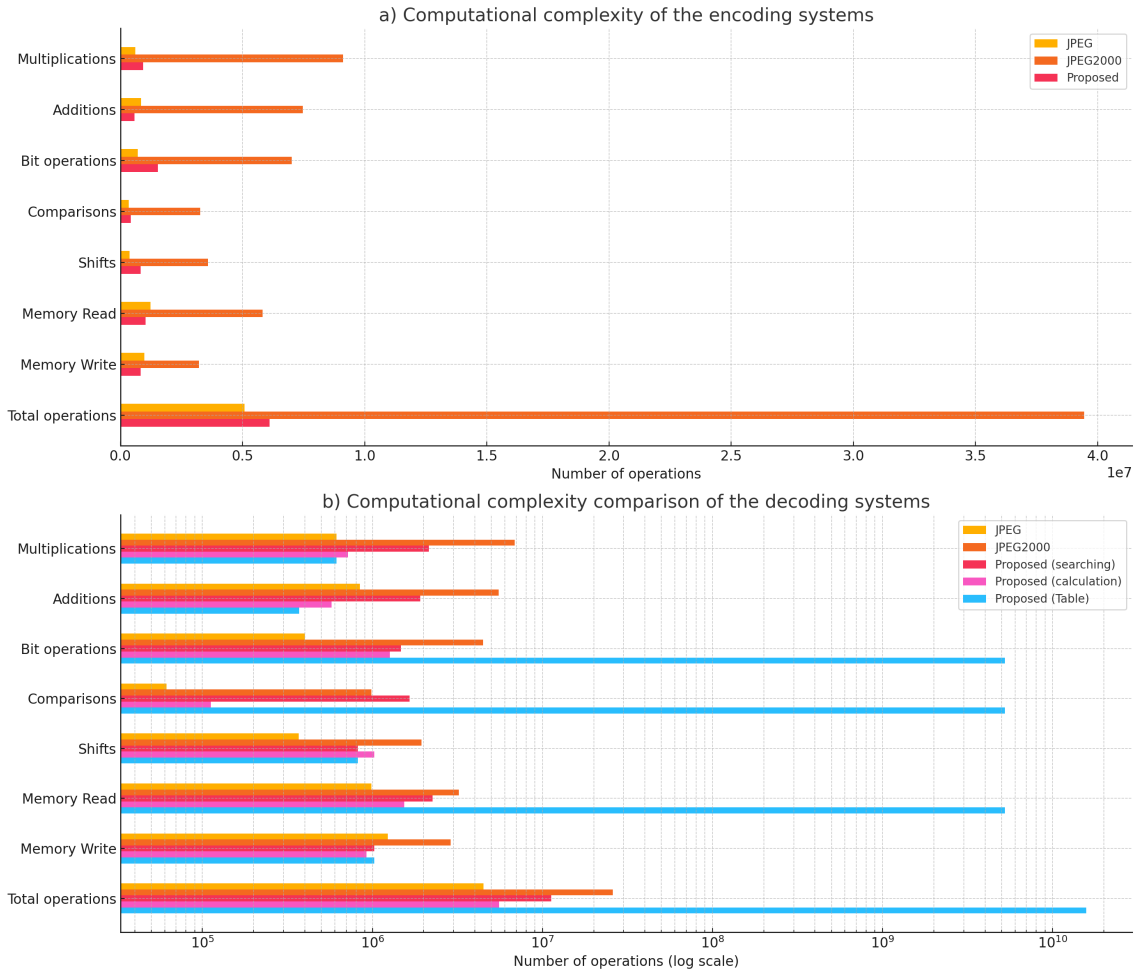


Figure 5.4: Comparison of the computational complexity of all implemented systems.

5.4 Thesis IV.

I introduced a centralized edge-assisted architecture for multi-robot Visual SLAM that overcomes the computational and resource limitations of fully onboard deployment. The framework strategically partitions tasks between robots and an edge device, assigning real-time tracking and local mapping to robots, while offloading global optimization, map fusion, and loop closure to the edge.

Supporting Statement:

The thesis was validated on a two-robot system. The approach achieves robust collaborative SLAM with consistent global map fusion, while reducing memory usage by up to 39% per robot and 34% overall at the edge. The results demonstrate

an efficient and scalable solution for multi-robot SLAM in bandwidth-constrained environments, supporting real-world autonomous deployments.

Related Publications: [P4], [P5], [P2].

5.5 Chapter Findings

The chapter presented an end-to-end framework architecture for a multi-robot V-SLAM system. It presented an ORB-SLAM multi-robot architecture and communication paradigm for efficient data exchange between the mobile robots and the centralized edge. In the proposed architecture, mobile robots handle tasks in real-time with low computational loads. The edge device processes computationally intensive tasks, enabling better map optimization and drift correction compared to a JPEG- and JPEG2000-based architecture. In addition, the compression module reduces bandwidth usage for transmitting data and ensures efficient and low-latency communication.

Chapter 6

Efficient FPGA Implementation of a Visual Data Encoding/Decoding Pipeline

6.1 Introduction and Literature Background

In practice, mobile robotic platforms are limited by restricted computation, limited communication bandwidth to edge or cloud systems, and strict real-time requirements. To address these limitations, Field-Programmable Gate Array (FPGA)-based solutions have emerged as an effective approach for accelerating image processing tasks. By exploiting parallelism and reconfigurability, FPGAs can achieve high-throughput while maintaining adaptability, making them suitable for real-time applications [109]. Although FPGA-based compression methods have been explored, many lack integrated with adaptive entropy coding and comprehensive system-level validation. In parallel, existing SLAM-oriented compression approaches often rely on software implementations, limiting their efficiency. This highlights the need for integrated hardware-accelerated solutions.

Recent research has explored various approaches to improve visual data processing under edge constraints, including transform-based compression, learned image compression, hardware acceleration, and lightweight encoding techniques. Learned image compression methods have demonstrated remarkable rate-distortion performance by employing neural-network-based coding frameworks [110, 111]. Furthermore, sev-

eral studies have investigated machine-oriented learned compression and FPGA deployment strategies that utilize fixed-point arithmetic, quantization-aware optimization, knowledge distillation, fine-grained pipelining, and algorithm-architecture co-optimization to improve throughput and energy efficiency while reducing the computational complexity of neural codecs [31, 112, 113, 114, 115, 116, 117]. Although these approaches have demonstrated the feasibility of deploying learned image compression on FPGA platforms, they generally rely on pretrained neural models, dedicated inference accelerators, and hardware-specific optimization strategies, resulting in considerably higher implementation complexity than deterministic transform-domain compression methods. Consequently, transform-based methods remain attractive for resource-constrained robotic systems because of their computational efficiency, deterministic execution, and straightforward FPGA implementation.

Methods based on Discrete Cosine Transform and JPEG2000 remain widely used due to their compression efficiency, with parallel implementations achieving high throughput [62]. Graphics Processing Unit (GPU)-based acceleration further enhances performance but at the cost of increased power consumption [118]. Alternatively, lightweight and lossless schemes such as JPEG-LS and predictive coding offer reduced complexity, though they face challenges in pipeline efficiency and deterministic performance [119]. Other methods, including row-based compression, improve computational efficiency but often provide limited adaptability [120]. FPGA-based vision systems have also demonstrated high-performance real-time processing; however, they typically exclude integrated compression, leading to increased data transfer overhead [121]. Additionally, secure compression techniques, such as bitstream-level encryption, enhance data protection but introduce complexity and are not well optimized for real-time hardware deployment [122]. In general, existing solutions often treat compression, vision processing, and security as separate components, resulting in suboptimal system performance.

In this chapter, a complete hardware–software co-design is implemented on an FPGA platform, with a partitioned architecture between programmable logic and processing system components. The proposed system accelerates computationally intensive operations while maintaining flexibility for control and entropy coding. The implemen-

ted system is integrated and evaluated within an edge-assisted indoor navigation framework. It enables efficient visual data reduction while supporting edge-assisted real-time processing. The system integrates a Processing System (PS) with Programmable Logic (PL) acceleration, and leverages a remote edge server to perform VSLAM in a bandwidth-efficient and latency-aware manner. Furthermore, it is validated within an edge-assisted VSLAM framework, enabling end-to-end evaluation of compression efficiency, navigation accuracy, and real-time requirements. This transition from software-based validation to a fully implemented embedded FPGA system represents the primary contribution of this chapter.

6.2 System Design and Implementation

This work proposes a hardware-software partitioning strategy for the compression pipeline between the Programmable Logic (PL) fabric and Processing System (PS) Central Processing Unit (CPU) capabilities for concurrent navigation tasks. This partition satisfies throughput constraints and resource budget limits of mobile robots. This section explains the partition and details the implementation of both portions.

6.2.1 Hardware-Software Co-Design Strategy

The partitioning of processing units between the programmable logic and the processing system is determined by three key considerations: the regularity of computation, the degree of data parallelism, and control flow complexity. Processing units that exhibit deterministic behavior, operate on fixed-size data structures, and involve highly parallel arithmetic operations are well-suited for implementation in the PL. In these cases, High-Level Synthesis (HLS) tools can effectively leverage loop unrolling, pipelining, and array partitioning, thereby enabling high-throughput execution with minimal control overhead. In contrast, units that depend on irregular computation patterns, complex control mechanisms, or serial computational behaviors are more appropriately assigned to the PS. This includes components that rely on dynamic data structures, such as hash tables used for dictionary-based operations, as well as modules that implement probabilistic models, including range-coded state machines.

6.3 Encoding and Decoding Pipeline

The encoding pipeline processes incoming RGB keyframe blocks through a sequence of operations, starting with the candidate keyframe received from the tracking unit and ending with the generation of the final compressed data. Hardware-accelerated units are integrated as Vitis HLS generated Advanced eXtensible Interface version 4 (AXI4) Intellectual Property (IP) cores imported to AMD Vivado design tool, while entropy coding and protocol framing are executed on the Arm Cortex-A53 PS and implemented using AMD Vitis Integrated Development Environment (IDE). The proposed end to end encoding system can be illustrated in the block diagram shown in Figure E.1. The system architecture supports an analyze-then-compress operational paradigm within the VSLAM framework, enabling compressed keyframe transmission that significantly reduces wireless bandwidth consumption while preserving navigation accuracy measured by Trajectory Root Mean Square Error (RMSE) and image fidelity metrics; Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index Measure (SSIM).

The proposed compression pipeline consists of block-based processing, DCT transformation, quantization, zigzag scanning, DC differential encoding, AC sparsity exploitation, bit packing, dictionary encoding, and range entropy coding. This multistage approach enables efficient data reduction while preserving essential visual information. The PL side accelerates computationally intensive operations, including DCT, quantization, zigzag scanning, and bit packing, while the PS handles dictionary mapping and entropy coding. This hardware-software partition balances performance, flexibility, and resource efficiency. Input keyframes are first preprocessed by being divided into non-overlapping 8×8 pixel blocks, and a 2D DCT is applied to each block. A direct floating-point implementation is not suitable for real-time FPGA deployment due to high resource consumption and variable latency. Instead, a prescaled fixed-point approximation is employed to enable efficient hardware realization while maintaining acceptable numerical accuracy. The separability of the 2D DCT is exploited by decomposing it into two sequential one-dimensional transforms (row-wise followed by column-wise), each operating on an 8-element fixed-point vector.

This significantly reduces computational complexity and enables efficient pipelined processing. Figure 6.1 illustrates the implementation of the 2D DCT as two consecutive 1D DCT operations applied in the horizontal and vertical directions.

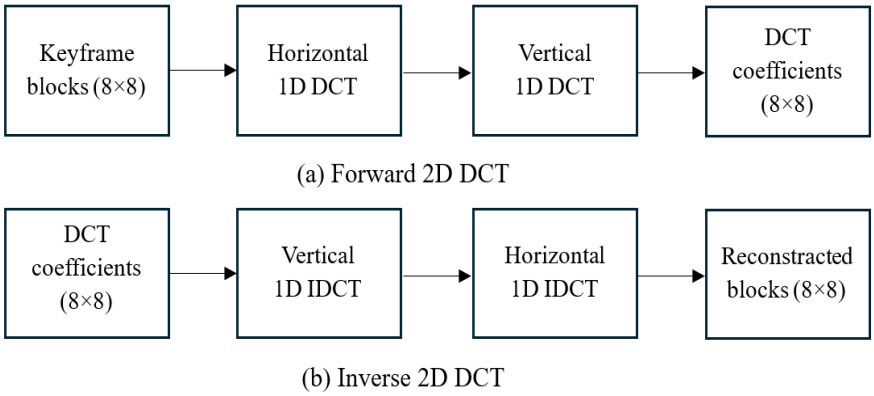


Figure 6.1: Decomposition of the 2D DCT into horizontal and vertical 1D DCT stages.

The resulting DCT coefficients are quantized and rounded to the nearest integer. A key architectural feature of this work is the runtime-programmable quantizer, where the quantization matrix can be updated via the AXI4-Lite interface without requiring FPGA re-synthesis. This allows dynamic adjustment of compression quality in response to varying network conditions and real-time processing requirements. For instance, in feature-rich environments, a coarser quantizer may be applied to maintain throughput, whereas finer quantization can be used in critical regions requiring higher localization accuracy. Following quantization, zigzag scanning is applied to reorder the coefficients into a one-dimensional sequence, starting with the DC component, followed by AC coefficients arranged from low to high frequency. This ordering enhances compression efficiency by increasing the likelihood of consecutive zero-valued coefficients. In the PL datapath, the zigzag permutation is implemented using a static lookup table (LUT) mapped to the BRAM address sequencer, incurring no additional latency and minimal hardware overhead. To further exploit spatial redundancy, differential pulse code modification (DPCM) is used to encode the DC coefficients by representing the difference between adjacent blocks, thereby reducing signal variance. For AC coefficients, a two-level representation is used. First, a 63-bit binary mask identifies the positions of non-zero AC coefficients, encoded as a

compact 64-bit word including the DC position. Second, only the non-zero AC values are stored in the coefficient stream in zigzag order, omitting zero-valued entries entirely. The PL datapath then packs every three coefficients into a single word using a fixed-field encoding scheme, producing a compact data stream that can be efficiently reconstructed at the edge server. The compression efficiency depends on coefficient sparsity; for example, a block containing 12 non-zero coefficients can be represented using only four packed words instead of the original 64 values, achieving significant data reduction before entropy coding. Finally, the packed data stream is transferred to the PS, where dictionary-based encoding and range entropy coding are applied to produce the final compressed output.

On the PS side, dictionary-based symbol substitution is applied at the edge to encode the packed data stream more efficiently. Empirical observations from indoor navigation datasets indicate that only a limited number of packed word values occur frequently, while the majority appear rarely. Based on this, a frequency-ordered dictionary, referred to as the Dictionary Table (DICT), is constructed, where entries are arranged according to their occurrence frequency to support fast lookup. Common patterns are represented using compact indices, whereas infrequent symbols are encoded using an escape (ESC) mechanism. The resulting index stream is then processed by a binary-scaled integer range coder to generate a compressed bitstream. During decompression, the procedure is reversed: the range decoder first reconstructs the index sequence, which is then translated back into the original packed symbols through inverse dictionary mapping. This approach provides effective entropy coding while ensuring deterministic reconstruction of the data. Communication between the PL and the PS is implemented using a dual AXI architecture for each PL IP core. The AXI4-Lite slave interface is used for control and configuration, including parameters such as addresses, encoding keys, and shift offsets, as well as status signaling. In contrast, the AXI4 full master interface supports high throughput data transfer through burst Direct Memory Access (DMA) transactions. Larger data structures, such as image blocks, DCT coefficients, quantization matrices, and packed output streams, are transferred using 256-bit AXI4 bursts, which reduces address overhead and improves overall memory bandwidth utilization. The architecture of the

FPGA-based system implemented is illustrated through the block diagram shown in Figure E.2.

6.4 Implementation results and Discussion

The proposed system was evaluated on the AMD Kria KV260 Vision AI Starter Kit (XCK26 UltraScale+ MPSoC) platform against conventional image compression standards, including JPEG and JPEG2000. The evaluation was conducted on the TUM RGB-D benchmark dataset, which accurately timestamps 640×480 RGB sequences from an Asus Xtion sensor captured in a diverse indoor environment including 2585 images captured at 30 fps. The ground truth trajectory is provided by the TUM benchmark motion capture system at sub-millimeter accuracy. The trajectory diameter of the benchmark is 5.12 m × 4.89 m × 0.54 m, and its duration is 87.09 s. Notably, the trajectory forms a consistent loop, as the final segment closely aligns with the starting position, thereby enabling reliable evaluation of loop closure performance.

Evaluation metrics used in this paper include PSNR, SSIM, and processing latency. While the primary objective of this work is to achieve an efficient deployment of the encoder on the mobile robot, the decoder is intended to be implemented on an edge server on a CPU or any available platform. However, the decoder is implemented using a hardware–software co-design approach with FPGA-based acceleration, enabling a symmetric FPGA-to-FPGA architecture between the robot and the edge server, and to provide a consistent evaluation. This symmetry ensures compatibility in data representation and processing, while also improving throughput and reducing latency during reconstruction. By offloading the computationally demanding decoding tasks to an FPGA-accelerated edge platform, the overall system maintains a balanced distribution of workload, enhancing performance without imposing additional constraints on the resource-limited mobile robot. The keyframe encoding and decoding units are implemented using fixed point arithmetic and deployed on the FPGA, and compared to the JPEG and JPEG2000 software versions implemented using MATLAB-based computer CPU running on Intel Core i7 - 12700H processor

and 32 GB of RAM. The encoded/decoded visual data is provided separately to the ORB-SLAM to test the compression efficiency and navigation accuracy.

Tables 6.1 and 6.2 show the encoding results of a sample keyframe obtained from the mobile robot section using the standards of JPEG and JPEG2000 implemented on the CPU, respectively. Both present the compressed size, percentage Compression Ratio (CR%), PSNR, SSIM, in addition to encoding and decoding time measurements. The sample keyframe sequence tested within the dataset is 1132, and the keyframe size is 300 KB. Similarly, the proposed encoding and decoding are performed using the FPGA-based system. Table 6.3 shows the results.

Table 6.1: Encoding performance of the JPEG system (CPU-based implementation)

CR [%]	**Size** [**KB**]	**PSNR** [**dB**]	**SSIM**	**Enc.** [s]	**Dec.** [s]
68	153	34.81	0.9579	0.3457	0.5991
69	144	33.20	0.9330	0.2341	0.5303
70	141	32.35	0.9148	0.2279	0.5279
71	140	29.55	0.8842	0.2217	0.5071

Table 6.2: Encoding performance of the JPEG2000 system (CPU-based floating-point implementation)

CR [%]	**Size** [**KB**]	**PSNR** [**dB**]	**SSIM**	**Enc.** [s]	**Dec.** [s]
73	134	44.03	0.9823	0.110	0.440
82	89.8	41.43	0.9721	0.136	0.412
85	72	40.09	0.9647	0.112	0.444
91	41	37.49	0.9483	0.195	0.532
97	15	34.82	0.9279	0.192	0.492
98	8.9	29.53	0.8647	0.212	0.417

Regarding the CPU-based JPEG implementation, moderate compression ratios are achieved. However, as compression increases, the quality of the image is noticeably reduced. Although encoding time slightly improves with higher compression, decoding time remains relatively constant, indicating limited efficiency gains and reduced capability in preserving fine image details. In contrast, the CPU-based JPEG2000 implementation demonstrates significantly better performance across all metrics. It

Table 6.3: Encoding performance of the proposed method (FPGA-based fixed-point implementation)

CR [%]	**Size** [**KB**]	**PSNR** [**dB**]	**SSIM**	**Enc.** [s]	**Dec.** [s]
82	87	35.23	0.9515	0.3205	0.5241
86	68	33.55	0.9398	0.3174	0.5018
88	59	32.67	0.9328	0.3021	0.4888
89	54	32.21	0.9271	0.2874	0.4217
92	38	31.07	0.8961	0.2547	0.3913
93	32	30.15	0.8721	0.2112	0.3776
94	27	29.25	0.8501	0.1682	0.3358

achieves higher compression ratios while maintaining superior image quality. Even at high compression levels, the degradation in quality is gradual and remains acceptable until extreme compression is applied. Additionally, JPEG2000 shows faster encoding times and comparable decoding times relative to JPEG, highlighting the efficiency of its wavelet-based approach. The FPGA-based proposed method implementation, using fixed-point arithmetic, exhibits slightly lower PSNR and SSIM values compared to JPEG2000, reflecting the impact of reduced numerical precision. Nevertheless, it maintains acceptable visual quality and demonstrates more consistent and often improved decoding times, along with steadily decreasing encoding times as compression increases. This indicates effective hardware acceleration and suitability for real-time processing. In summary, JPEG2000 clearly outperforms JPEG in both compression efficiency and image quality. The FPGA-based implementation provides the best overall performance, it offers a practical trade-off by slightly reducing quality in data exchange for improved computational efficiency and real-time applicability.

The relationship between RMSE of the trajectory and the compression ratio is given in Figure 6.2, which demonstrates that RMSE increases as the compression ratio increases. This indicates that stronger compression leads to greater reconstruction error and reduced accuracy. For JPEG, the curve is limited to lower compression ratios (around 68–71%) and shows a gradual increase in RMSE. Even within this narrow range, the error rises noticeably, suggesting that JPEG is more sensitive to compression and less efficient in preserving trajectory accuracy. In contrast, JPEG2000

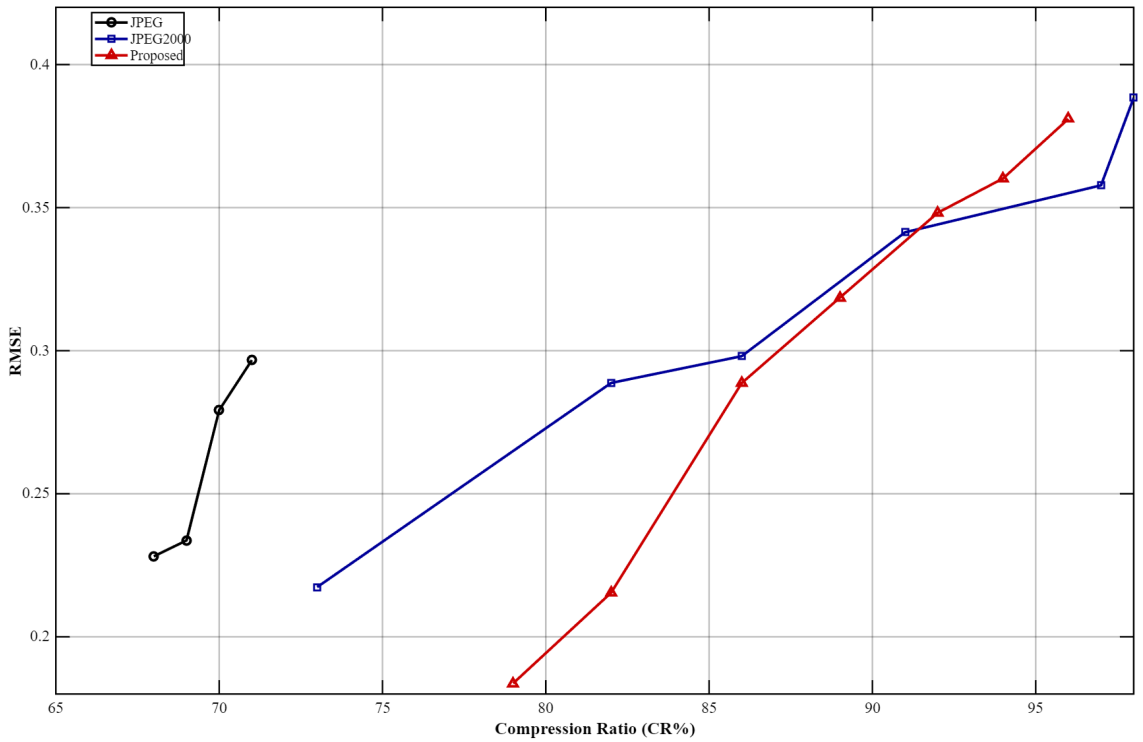
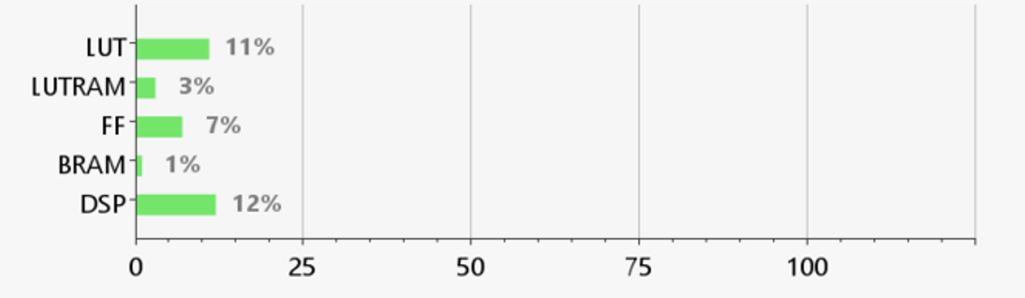


Figure 6.2: Graph of RMSE for VSLAM trajectory vs. compression ratio of the three implemented encoding systems.

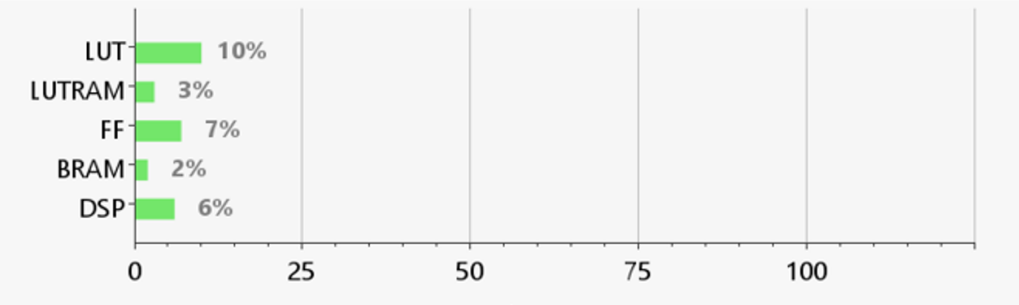
exhibits a smoother and more extended trend across a wider range of compression ratios. While RMSE increases steadily with CR%, it remains lower than JPEG at comparable points, demonstrating better robustness and reconstruction quality under higher compression. The proposed FPGA-based method follows a similar upward trend but generally achieves lower RMSE than JPEG2000 at moderate compression levels (around 79–89%). This indicates improved accuracy in this range. However, at very high compression ratios (above 90%), the RMSE increases more rapidly and approaches or slightly exceeds that of JPEG2000, reflecting a trade-off between compression efficiency and precision.

Figures 6.3 and 6.4 illustrate the resource utilization and power consumption of the proposed FPGA-based encoder and decoder.

As shown in Figure 6.3, both designs exhibit low resource utilization, confirming the efficiency of the architecture. The encoder requires slightly higher resources, particularly in Digital Signal Processing (DSP) slice usage (12%) and LUTs (11%), while the decoder shows reduced DSP utilization (6%) with similar usage of LUTs

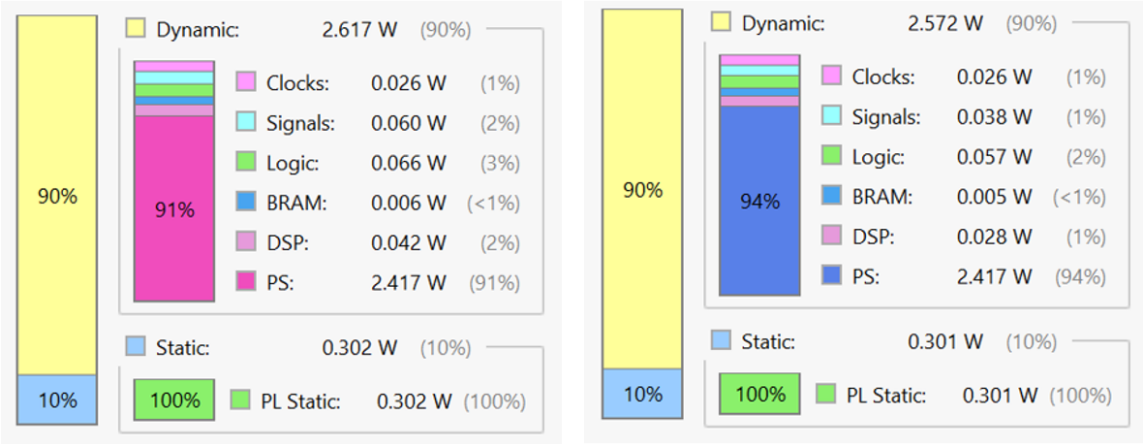


(A) Encoder circuit (robot side)



(B) Decoder circuit (edge side)

Figure 6.3: Resource utilization of the implemented FPGA-based system.



(A) Encoder (robot side)

(B) Decoder (edge side)

Figure 6.4: Power consumption of the implemented FPGA-based system.

(10%) and flip-flops (7%). Overall, the low utilization across LUTs, BRAM, and logic elements indicates that the design is lightweight and suitable for embedded implementations with available hardware margin. Figure 6.4 shows that the total power consumption is approximately 2.9 W for both modules, with dynamic power

accounting for about 90% and static power around 10%. The processing system dominates power consumption, contributing more than 90% of the dynamic power, while the programmable logic consumes only a small portion. The decoder exhibits slightly lower dynamic power, consistent with its reduced computational demand.

6.5 Thesis V.

I proposed a hardware-software co-design framework for visual data compression, implemented on an FPGA-based platform with an optimized partitioning strategy between the Programmable Logic (PL) and Processing System (PS).

Supporting Statement:

This contribution addresses the critical challenge of deploying high-performance visual compression in resource constrained robotic systems, effectively bridging the gap between algorithmic design and practical hardware realization for machine vision applications.

Related Publications: [P1], [P9], [P10].

6.6 Chapter Findings

This chapter presents a complete FPGA-based visual compression pipeline for edge-assisted indoor navigation systems. The framework uses fixed-point arithmetic within an efficiently designed PL–PS partitioning scheme, resulting in low hardware utilization and balanced power consumption. Experimental results indicate that the system preserves competitive reconstruction quality at high compression ratios while supporting real-time operation on embedded platforms. These results confirm the effectiveness of the hardware–software co-design approach and demonstrate its suitability for resource-constrained robotic applications.

Chapter 7

Summary and Future Research Directions

7.1 Dissertation Summary

The rapid advancement of autonomous robotic systems and machine vision applications has intensified the demand for efficient visual data processing under strict computational, energy, storage, and bandwidth constraints. This dissertation addresses these challenges by proposing a unified framework for efficient visual data encoding methods that bridges the domains of 2D image compression, 3D reconstruction, and VSLAM. The core problem lies in achieving high compression ratios while preserving the structural and feature-level information necessary for accurate spatial perception, including feature extraction, pose estimation, and geometric reconstruction. To address this problem, the dissertation introduces a series of novel encoding and compression strategies. A multi-level visual data encoding scheme is developed and tightly integrated into monocular ORB-SLAM systems. This approach optimizes the trade-off between compression efficiency and feature preservation, enabling robust localization and mapping under varying quantization levels and limited bandwidth conditions. The proposed methods demonstrate superior performance compared to the conventional JPEG and JPEG2000 standards in maintaining trajectory accuracy and reconstruction quality.

The dissertation also introduces an edge-assisted architecture for multi-robot VSLAM, where computational tasks are efficiently partitioned between mobile agents and cent-

ralized edge devices. By incorporating a proposed visual data encoding and communication module, the framework significantly reduces memory usage and communication overhead while enabling collaborative map fusion and consistent global localization across multiple robots. The dissertation further extends visual data compression techniques to the domain of 3D reconstruction. A novel DCT-based transform-domain encoding method, enhanced with optimized quantization, coefficient reduction, and matrix reduction strategy, is proposed to compress 2D structured-light and multi-view images. These methods achieve high compression ratios while preserving critical geometric details required for accurate 3D mesh reconstruction, overcoming limitations of traditional JPEG-based approaches.

In addition, the dissertation investigates an alternative transform-based encoding, including Discrete Fourier Transform (DFT) combined with proposed techniques, demonstrating that high-frequency coefficient encoding can be effectively exploited for high-efficiency compression without compromising reconstruction fidelity. The support of decoding algorithms further improves the speed and practicality of reconstruction for real-time systems. The findings of this dissertation establish that visual data encoding is not merely a pre-processing step but a critical component that directly influences the robustness and scalability of spatial perception systems. By unifying compression strategies with perception pipelines, the proposed methods enable efficient, accurate, and scalable VSLAM and 3D reconstruction in bandwidth-constrained and resource-limited environments.

Finally, the dissertation presents a complete FPGA-based hardware-software co-design framework for a unified visual data compression system. The proposed architecture demonstrated efficient resource utilization, low power consumption, and real-time performance, validating its suitability for embedded robotic perception systems.

7.2 Future Research Directions

The results of this dissertation open up several directions for further research in visual data encoding, edge-assisted VSLAM, 3D reconstruction, and FPGA-based

machine vision systems. An important direction is to extend the proposed compression and encoding methods beyond the evaluated datasets and apply them to more diverse robotic scenarios, such as under different motion patterns, lighting conditions, camera viewpoints, and communication constraints. Therefore, further validation would help assess the generality of the proposed methods and their robustness in real-world environments. The proposed methods can be extended further to adaptive architectures in which compression level, keyframe selection, and task partitioning are adjusted dynamically according to network bandwidth, robot velocity, available energy, and mapping uncertainty. This would make the encoding process more responsive to the actual operating conditions than relying on fixed compression settings.

The multi-robot VSLAM architecture can be extended toward larger and more heterogeneous robotic teams. Future work may investigate scalable map fusion, distributed loop closure, and communication-aware coordination among robots equipped with different sensors and computational capabilities.This direction could be explored for applications such as warehouse automation, search-and-rescue operations, agricultural monitoring, and collaborative industrial inspection.

For 3D reconstruction applications, future research may apply the proposed DCT, DFT, and coefficient reduction-based compression methods to broader vision perception domains, such as industrial surface inspection, digitization of cultural heritage, medical 3D imaging, robotic grasping, and quality control.

In the same context, future research may include a controlled experimental comparison between the proposed transform-domain encoding methods and representative learned image compression models. Such comparison should be performed under identical robotic perception conditions, including the same dataset, image resolution, bitrate or compression-ratio operating points, hardware platform, inference framework, and downstream perception metrics such as trajectory RMSE, loop-closure reliability, mapping consistency, and 3D reconstruction accuracy. This direction would allow a fair assessment of the trade-off between the strong rate-distortion performance of learned codecs and the deterministic behavior, lower computational

complexity, and FPGA suitability of transform-domain methods. Lightweight learning modules can be used to select quantization parameters, predict important regions, or control encoding levels, while keeping the core compression pipeline simple, explainable, and hardware-friendly.

Recent studies in embedded SLAM and FPGA-based robotic computing show increasing interest in accelerating front-end perception tasks such as feature extraction, feature matching, and 3D reconstruction which supports the relevance of extending the proposed hardware-software co-design toward a complete perception pipeline.

7.3 Theses

7.3.1 Thesis I.

I introduced a novel image compression algorithm based on the 2D Discrete Fourier Transform (2D DFT) with an enhanced coefficients reduction algorithm and zero elimination technique, achieving high compression ratios while preserving image quality.

Supporting Statement:

This contribution addresses a fundamental limitation of DFT-based compression, namely the difficulty of efficiently compressing real and imaginary components without degrading the fidelity of the reconstruction. By integrating DFT with the proposed minimization strategy, the method overcomes this constraint and achieves a high-compression efficiency while preserving reconstruction quality, thereby outperforming the traditional JPEG standard.

Related Publications: [P15], [P11], [P12].

7.3.2 Thesis II.

I proposed a novel lossy image compression algorithm optimized for 3D mesh reconstruction from structured-light and multi-view images, addressing the limitations of JPEG at high compression ratios. The pro-

posed method improves DCT-based compression by optimizing the quantization matrix and introducing an efficient coefficient management pipeline, including DC differencing, zero-elimination, and arithmetic coding.

Supporting Statement:

The approach achieves up to 99% compression while preserving reconstruction quality, maintaining accurate geometry and surface details, where JPEG fails beyond 97% compression. This contribution provides an efficient solution for storage and transmission in 3D reconstruction pipelines, significantly reducing data requirements without sacrificing geometric fidelity.

Related Publications: [P13], [P14], [P8].

7.3.3 Thesis III.

I developed and implemented a novel multi-level encoding method for visual data compression specifically tailored for robust monocular ORB-SLAM systems operating in cloud-edge computing environments. This comprehensive approach addresses the critical challenge of bandwidth optimization in visual SLAM applications while maintaining trajectory estimation accuracy and feature detection quality.

Sub-thesis III.1

I have designed and validated two distinct multi-level encoding schemes: a single-level implementation and a two-level implementation, both integrating seamlessly with the ORB-SLAM architecture.

Sub-thesis III.2

I have demonstrated that the proposed multi-level encoding method significantly outperforms traditional JPEG and JPEG2000 compression in terms of robustness under varying quantization factors, making it more suitable for resource-constrained mobile devices.

Related Publications: [P3], [P6], [P7].

7.3.4 Thesis IV.

I introduced a centralized edge-assisted architecture for multi-robot Visual SLAM that overcomes the computational and resource limitations of fully onboard deployment. The framework strategically partitions tasks between robots and an edge device, assigning real-time tracking and local mapping to robots, while offloading global optimization, map fusion, and loop closure to the edge.

Supporting Statement:

The thesis was validated on a two-robot system. The approach achieves robust collaborative SLAM with consistent global map fusion, while reducing memory usage by up to 39% per robot and 34% overall at the edge. The results demonstrate an efficient and scalable solution for multi-robot SLAM in bandwidth-constrained environments, supporting real-world autonomous deployments.

Related Publications: [P4], [P5], [P2].

7.3.5 Thesis V.

I proposed a hardware-software co-design framework for visual data compression, implemented on an FPGA-based platform with an optimized partitioning strategy between the Programmable Logic (PL) and Processing System (PS).

Supporting Statement:

This contribution addresses the critical challenge of deploying high-performance visual compression in resource constrained robotic systems, effectively bridging the gap between algorithmic design and practical hardware realization for machine vision applications.

Related Publications: [P1], [P9], [P10].

Author's publications

[P1] O. M. Salih and J. Vásárhelyi, 'A hardware–software fpga co-design of a visual data compression pipeline for indoor edge-assisted navigation,' in *2026 27th International Carpathian Control Conference (ICCC)*, IEEE, 2026, pp. 1–6.

Scopus indexed.

[P2] H. Rostum, O. M. Salih and J. Vásárhelyi, 'Efficient real-time drone mapping with visual slam and python,' *Pollack Periodica*, pp. 606–2025. 2026,

Scopus [Q3].

[P3] O. M. Salih and J. Vásárhelyi, 'End-to-end multi-level encoding methods of visual data compression for robust monocular visual orb-slam,' *Acta Polytechnica Hungarica*, vol. 22, no. 5, 2025,

Scopus [Q2], WoS (SCIE) [Q2].

[P4] O. M. Salih and J. Vásárhelyi, 'Efficient architecture for multi-robot visual slam systems,' in *2025 26th International Carpathian Control Conference (ICCC)*, IEEE, 2025, pp. 1–6.

Scopus indexed.

[P5] V. József and S. M. Omar, 'Peremhálózat alapú vslam keretrendszer erőforrás-korlátozott platformokhoz: An edge assisted vslam framework for resource constraint platforms,' *Energetika-Elektrotechnika–Számítástechnika és Oktatás Multikonferencia*, pp. 178–182, 2025.

[P6] O. M. Salih, H. Rostum and J. Vásárhelyi, 'A novel method to improve the efficiency and performance of cloud-based visual simultaneous localization and mapping,' *Engineering Proceedings*, vol. 79, no. 1, p. 78, 2024,

Scopus [Q3].

[P7] O. M. Salih and J. Vásárhelyi, 'Visual data compression approaches for edge-based orb-vslam systems,' in *2024 25th International Carpathian Control Conference (ICCC)*, IEEE, 2024, pp. 1–6.

Scopus indexed.

[P8] O. M. Salih and J. Vásárhelyi, 'Visual data compression for machine vision applications,' in *Doktoranduszok fóruma: Gépészmérnöki és Informatikai Kar szekciókiadványa Conference : Miskolc Egyetemváros, Hungary 2023.11.23. - 2023,11,23*, University of Miskolc, 2024, pp. 128–133.

[P9] J. Vásárhelyi, O. M. Salih, H. M. Rostum and R. Benotsname, 'An overview of energies problems in robotic systems,' *Energies*, vol. 16, no. 24, p. 8060, 2023,
Scopus [Q1], WoS (SCIE) [Q1].

[P10] M. R. Hussam, M. S. Omar and H. Noha, 'Fpga implementation in mobile robot applications: State of the art review,' *Multidiszciplináris Tudományok*, vol. 13, no. 2, pp. 232–249, 2023.

[P11] O. M. Salih, H. M. Rostum, J. Vásárhelyi and M. M. Siddeq, 'Fast joint image compression-encryption algorithm used for 3d reconstruction,' in *2023 24th International Carpathian Control Conference (ICCC)*, IEEE, 2023, pp. 400–405.
Scopus indexed.

[P12] O. M. Salih, 'Quick sequential search algorithm used to decode high frequency matrices,' *International Journal of Electrical and Computer Engineering*, 2023.

[P13] O. M. Salih, M. H. Rasheed, M. M. Siddeq and M. A. Rodrigues, 'Image compression for quality 3d reconstruction,' *Journal of King Saud University-Computer and Information Sciences*, vol. 34, no. 5, pp. 2271–2287, 2022,
Scopus [Q1], WoS (SCIE) [Q1].

[P14] M. H. Rasheed, O. M. Salih and M. M. Siddeq, 'Joint image encryption and compression schemes based on hexa-coding,' *Periodicals of Engineering and Natural Sciences*, vol. 9, no. 2, pp. 569–580. 2021,
Scopus [Q2].

[P15] M. H. Rasheed, O. M. Salih, M. M. Siddeq and M. A. Rodrigues, 'Image compression based on 2d discrete fourier transform and matrix minimization algorithm,' *Array*, vol. 6, p. 100 024, 2020,
WoS (ESCI) [Q1].

References

[1] C. R. Steffens, L. R. V. Messias, P. L. J. Drews and S. S. da Costa Botelho, 'Can exposure, noise and compression affect image recognition? an assessment of the impacts on state-of-the-art convnets,' in *2019 Latin American Robotics Symposium (LARS), 2019 Brazilian Symposium on Robotics (SBR) and 2019 Workshop on Robotics in Education (WRE)*, IEEE, 2019, pp. 61–66.

[2] I. Bouderbal, A. Amamra and M. A. Benatia, 'How would image downsampling and compression impact object detection in the context of self-driving vehicles?' In *International Conference on Computing Systems and Applications*, Springer, 2020, pp. 25–37.

[3] A. Junaedy, H. Masuta, K. Sawai, T. Motoyoshi and N. Takagi, 'Real-time 3d map building in a mobile robot system with low-bandwidth communication,' *Robotics*, vol. 12, no. 6, p. 157, 2023.

[4] G. Alex and Y. Schechner Yoav, 'Image compression optimized for 3d reconstruction by utilizing deep neural networks [j],' *Journal of Visual Communication and Image Representation*, vol. 79, 2021.

[5] Y. M. Mostafa, M. N. Al-Berry and H. A. Shedeed, '3d reconstruction from jpg images.,' *International Journal of Advanced Computer Science & Applications*, vol. 16, no. 7, 2025.

[6] C.-Y. Wu, J. Johnson, J. Malik, C. Feichtenhofer and G. Gkioxari, 'Multiview compressive coding for 3d reconstruction,' in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 9065–9075.

[7] M. Dejean-Servières, K. Desnos, K. Abdelouahab, W. Hamidouche, L. Morin and M. Pelcat, 'Study of the impact of standard image compression techniques on performance of image classification with a convolutional neural network,' Ph.D. dissertation, INSA Rennes; Univ Rennes; IETR; Institut Pascal, 2017.

[8] Z. Liu et al., 'Deepn-jpeg: A deep neural network favorable jpeg-based image compression framework,' in *Proceedings of the 55th annual design automation conference*, 2018, pp. 1–6.

[9] Y.-Y. Jo et al., 'Impact of image compression on deep learning-based mammogram classification,' *Scientific Reports*, vol. 11, no. 1, p. 7924, 2021.

[10] F. Ghazvinian Zanjani, S. Zinger, B. Piepers, S. Mahmoudpour, P. Schelkens and P. H. de With, 'Impact of jpeg 2000 compression on deep convolutional neural networks for metastatic cancer detection in histopathological images,' *Journal of Medical Imaging*, vol. 6, no. 2, pp. 027 501–027 501, 2019.

[11] Z. Liu et al., 'Machine vision guided 3d medical image compression for efficient transmission and accurate segmentation in the clouds,' in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 12 687–12 696.

[12] E.-H. Yang, H. Amer and Y. Jiang, 'Compression helps deep learning in image classification,' *Entropy*, vol. 23, no. 7, p. 881, 2021.

[13] G. Hamano, S. Imaizumi and H. Kiya, 'Effects of jpeg compression on vision transformer image classification for encryption-then-compression images,' *Sensors*, vol. 23, no. 7, p. 3400, 2023.

[14] K. Grm, V. Štruc, A. Artiges, M. Caron and H. K. Ekenel, 'Strengths and weaknesses of deep learning models for face recognition against image degradations,' *Iet Biometrics*, vol. 7, no. 1, pp. 81–89, 2018.

[15] T. Benbarrad, L. Eloutouate, M. Arioua, F. Elouaai and M. D. Laanaoui, 'Impact of image compression on the performance of steel surface defect classification with a cnn,' *Journal of Sensor and Actuator Networks*, vol. 10, no. 4, p. 73, 2021.

[16] H. Xiong and Y. Xu, 'Saliency-guided learned image compression for object detection,' in *International Conference on Neural Information Processing*, Springer, 2022, pp. 324–335.

[17] J. Zhong, T. Lei, G. Yao and P. Jiang, 'Robust aircraft detection with a simple and efficient model,' *Information*, vol. 9, no. 4, p. 74, 2018.

[18] E. S. Paulo, L. A. Thomaz, L. M. Távora, P. A. Assuncao and S. M. Faria, 'Extending the compression range of biomedical images for machine vision analysis,' in *2022 30th European Signal Processing Conference (EUSIPCO)*, IEEE, 2022, pp. 1273–1277.

[19] B. Cardone and F. Di Martino, 'Bit reduced fcm with block fuzzy transforms for massive image segmentation,' *Information*, vol. 11, no. 7, p. 351, 2020.

[20] W. Yang, H. Huang, Y. Hu, L.-Y. Duan and J. Liu, 'Video coding for machines: Compact visual representation compression for intelligent collaborative analytics,' *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 46, no. 7, pp. 5174–5191, 2024.

[21] G. K. Wallace, 'The jpeg still picture compression standard,' *Communications of the ACM*, vol. 34, no. 4, pp. 30–44, 1991.

[22] D. S. Taubman, M. W. Marcellin and M. Rabbani, 'Jpeg2000: Image compression fundamentals, standards and practice,' *Journal of Electronic Imaging*, vol. 11, no. 2, pp. 286–287, 2002.

[23] G. J. Sullivan, J.-R. Ohm, W.-J. Han and T. Wiegand, 'Overview of the high efficiency video coding (hevc) standard,' *IEEE Transactions on circuits and systems for video technology*, vol. 22, no. 12, pp. 1649–1668, 2012.

[24] B. Bross et al., 'Overview of the versatile video coding (vvc) standard and its applications,' *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 10, pp. 3736–3764, 2021.

[25] J. Ballé, D. Minnen, S. Singh, S. J. Hwang and N. Johnston, 'Variational image compression with a scale hyperprior,' *arXiv preprint arXiv:1802.01436*, 2018.

[26] D. Minnen, J. Ballé and G. D. Toderici, 'Joint autoregressive and hierarchical priors for learned image compression,' *Advances in neural information processing systems*, vol. 31, 2018.

[27] Z. Cheng, H. Sun, M. Takeuchi and J. Katto, 'Learned image compression with discretized gaussian mixture likelihoods and attention modules,' in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 7939–7948.

[28] J. Cao, X. Yao, H. Zhang, J. Jin, Y. Zhang and B. W.-K. Ling, 'Slimmable multi-task image compression for human and machine vision,' *IEEE Access*, vol. 11, pp. 29 946–29 958, 2023.

[29] G. Lu, X. Ge, T. Zhong, Q. Hu and J. Geng, 'Preprocessing enhanced image compression for machine vision,' *IEEE transactions on circuits and systems for video technology*, vol. 34, no. 12, pp. 13 556–13 568, 2024.

[30] L. Liu, Z. Hu, Z. Chen and D. Xu, 'Icmh-net: Neural image compression towards both machine vision and human vision,' in *Proceedings of the 31st ACM International Conference on Multimedia*, 2023, pp. 8047–8056.

[31] Y.-H. Chen, Y.-C. Weng, C.-H. Kao, C. Chien, W.-C. Chiu and W.-H. Peng, 'Transtic: Transferring transformer-based image compression from human perception to machine perception,' in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 23 297–23 307.

[32] K. Sayood, *Introduction to data compression*. Morgan Kaufmann, 2017.

[33] K. R. Rao and P. Yip, *Discrete cosine transform: algorithms, advantages, applications*. Academic press, 2014.

[34] R. C. Gonzalez, *Digital image processing*. Pearson education india, 2009.

[35] S. Yuan and J. Hu, 'Research on image compression technology based on huffman coding,' *Journal of Visual Communication and Image Representation*, vol. 59, pp. 33–38, 2019.

[36] P. Li and K.-T. Lo, 'Joint image encryption and compression schemes based on 16× 16 dct,' *Journal of Visual Communication and Image Representation*, vol. 58, pp. 12–24, 2019.

[37] M. A. Rodrigues, A. Robinson and A. Osman, 'Efficient 3d data compression through parameterization of free-form surface patches,' in *2010 International Conference on Signal Processing and Multimedia Applications (SIGMAP)*, IEEE, 2010, pp. 130–135.

[38] M. M. Siddeq and G. Al-Khafaji, 'Applied minimized matrix size algorithm on the transformed images by dct and dwt used for image compression,' *International Journal of Computer Applications*, vol. 70, no. 15, 2013.

[39] M. Siddeq and M. Rodrigues, 'A novel 2d image compression algorithm based on two levels dwt and dct transforms with enhanced minimize-matrix-size algorithm for high resolution structured light 3d surface reconstruction,' *3D Research*, vol. 6, no. 3, p. 26, 2015.

[40] M. Siddeq and M. Rodrigues, 'A new 2d image compression technique for 3d surface reconstruction,' in *Proceedings of 18th International Conference on Computers (part of CSCC'14). Recent advances in computer engineering series, 1 (22)*, World Scientific, Engineering Academy and Society (WSEAS), 2014, pp. 379–386.

[41] M. Siddeq and M. Rodrigues, 'A novel image compression algorithm for high resolution 3d reconstruction,' *3D Research*, vol. 5, no. 2, p. 7, 2014.

[42] M. Siddeq and M. Rodrigues, 'Applied sequential-search algorithm for compression-encryption of high-resolution structured light 3d data,' in *9th International Conference on Computer Graphics, Visualization, Computer Vision and Image Processing*, IADIS Press, 2015, pp. 195–202.

[43] M. M. Siddeq and M. A. Rodrigues, 'A novel high-frequency encoding algorithm for image compression,' *EURASIP Journal on Advances in Signal Processing*, vol. 2017, no. 1, p. 26, 2017.

[44] M. M. Siddeq and M. A. Rodrigues, 'Dct and dst based image compression for 3d reconstruction,' *3D Research*, vol. 8, no. 1, p. 5, 2017.

[45] Z. Wang, Y. Jiang and S. Chen, 'Image parallel block compressive sensing scheme using dft measurement matrix,' *Multimedia Tools and Applications*, vol. 82, no. 14, pp. 21 561–21 583, 2023.

[46] D. Milekic, T. Lukic and L. Krstanovic, 'Mathematical insight into the fourier transform for 2d image analysis,' *Technical Sciences*, no. 01-50, p. 295, 2025.

[47] W. Burger and M. J. Burge, *Digital image processing: An algorithmic introduction*. Springer Nature, 2022.

[48] M. M. Siddeq and M. A. Rodrigues, 'A novel hexa data encoding method for 2d image crypto-compression,' *Multimedia Tools and Applications*, vol. 79, no. 9, pp. 6045–6059, 2020.

[49] M. Rodrigues, M. Kormann, C. Schuhler and P. Tomek, 'Robot trajectory planning using olp and structured light 3d machine vision,' in *International Symposium on Visual Computing*, Springer, 2013, pp. 244–253.

[50] M. Rodrigues, M. Kormann, C. Schuhler and P. Tomek, 'Structured light techniques for 3d surface reconstruction in robotic tasks,' in *Proceedings of the 8th International Conference on Computer Recognition Systems CORES 2013*, Springer, 2013, pp. 805–814.

[51] M. Rodrigues, M. Kormann, C. Schuhler and P. Tomek, 'An intelligent real time 3d vision system for robotic welding tasks,' in *2013 9th International Symposium on Mechatronics and its Applications (ISMA)*, IEEE, 2013, pp. 1–6.

[52] A. Adler, D. Boublil and M. Zibulevsky, 'Block-based compressed sensing of images via deep learning,' in *2017 IEEE 19th International Workshop on Multimedia Signal Processing (MMSP)*, IEEE, 2017, pp. 1–6.

[53] M. Al-Ani and T. A. Hammouri, 'Video compression algorithm based on frame difference approaches,' *International Journal on Soft Computing*, vol. 2, no. 4, p. 67, 2011.

[54] A. Adler, 'Covariance-assisted matching pursuit,' *IEEE Signal Processing Letters*, vol. 23, no. 1, pp. 149–153, 2015.

[55] Y. Dar, M. Elad and A. M. Bruckstein, 'Optimized pre-compensating compression,' *IEEE Transactions on Image Processing*, vol. 27, no. 10, pp. 4798–4809, 2018.

[56] Y. J. Zhang, *Geometric modeling and mesh generation from scanned images*. Chapman and Hall/CRC, 2018.

[57] M. Botsch, L. Kobbelt, M. Pauly, P. Alliez and B. Lévy, *Polygon mesh processing*. CRC press, 2010.

[58] A. Maglo, G. Lavoué, F. Dupont and C. Hudelot, '3d mesh compression: Survey, comparisons, and emerging trends,' *ACM Computing Surveys (CSUR)*, vol. 47, no. 3, pp. 1–41, 2015.

[59] M. Rodrigues, A. Osman and A. Robinson, 'Partial differential equations for 3d data compression and reconstruction,' *ADSA Advances in Dynamical Systems and Applications*, vol. 8, no. 2, pp. 303–315, 2013.

[60] M. Siddeq and M. Rodrigues, '3d point cloud data and triangle face compression by a novel geometry minimization algorithm and comparison with other 3d formats,' in *Proceedings of the international conference on computational methods*, Scientech Publisher, vol. 3, 2016, pp. 379–394.

[61] M. M. Siddeq, 'Jpeg and sequential search algorithm applied on low-frequency sub-band,' *Journal of Information and Computing Science*, vol. 5, no. 3, pp. 163–172, 2010.

[62] D. Mukherjee, 'Parallel implementation of discrete cosine transform and its inverse for image compression applications: D. mukherjee,' *The Journal of Supercomputing*, vol. 80, no. 16, pp. 23 712–23 735, 2024.

[63] 3Dflow. '3df zephyr photogrammetry software,' Accessed: 2nd May 2026. [Online]. Available: `https://www.3dflow.net/3df-zephyr-photogrammetry-software/`.

[64] Autodesk. 'Autodesk marketplace application,' Accessed: 2nd May 2026. [Online]. Available: `https://marketplace.autodesk.com/apps/2e40e347-0369-4603-b83f-ecab324aed0f`.

[65] A. Tourani, H. Bavle, J. L. Sanchez-Lopez and H. Voos, 'Visual slam: What are the current trends and what to expect?' *Sensors*, vol. 22, no. 23, p. 9297, 2022.

[66] I. Vallivaara, J. Haverinen, A. Kemppainen and J. Röning, 'Magnetic field-based slam method for solving the localization problem in mobile robot floor-cleaning task,' in *2011 15th international conference on advanced robotics (ICAR)*, IEEE, 2011, pp. 198–203.

[67] Q. Zou, Q. Sun, L. Chen, B. Nie and Q. Li, 'A comparative analysis of lidar slam-based indoor navigation for autonomous vehicles,' *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 7, pp. 6907–6921, 2021.

[68] T. Yang, P. Li, H. Zhang, J. Li and Z. Li, 'Monocular vision slam-based uav autonomous landing in emergencies and unknown environments,' *Electronics*, vol. 7, no. 5, p. 73, 2018.

[69] Z. Liu et al., 'Real-time 6d lidar slam in large scale natural terrains for ugv,' in *2018 IEEE Intelligent Vehicles Symposium (IV)*, IEEE, 2018, pp. 662–667.

[70] G. Lu, H. Yang, J. Li, Z. Kuang and R. Yang, 'A lightweight real-time 3d lidar slam for autonomous vehicles in large-scale urban environment,' *IEEE Access*, vol. 11, pp. 12 594–12 606, 2023.

[71] B. Amjad, Q. Z. Ahmed, P. I. Lazaridis, M. Hafeez, F. A. Khan and Z. D. Zaharis, 'Radio slam: A review on radio-based simultaneous localization and mapping,' *IEEE Access*, vol. 11, pp. 9260–9278, 2023.

[72] A. Macario Barros, M. Michel, Y. Moline, G. Corre and F. Carrel, 'A comprehensive survey of visual slam algorithms,' *Robotics*, vol. 11, no. 1, p. 24, 2022.

[73] J.-C. Trujillo, R. Munguia, S. Urzua, E. Guerra and A. Grau, 'Monocular visual slam based on a cooperative uav–target system,' *Sensors*, vol. 20, no. 12, p. 3531, 2020.

[74] J. Mo, M. J. Islam and J. Sattar, 'Fast direct stereo visual slam,' *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 778–785, 2021.

[75] B. Li, Y. Guo, Z. Mi, Y. Yang and M. S. Obaidat, 'A novel data compression technique incorporated with computer offloading in rgb-d slam,' in *2019 International Conference on Computer, Information and Telecommunication Systems (CITS)*, IEEE, 2019, pp. 1–5.

[76] S. Eger, R. Pries and E. Steinbach, 'Evaluation of different task distributions for edge cloud-based collaborative visual slam,' in *2020 IEEE 22nd International Workshop on Multimedia Signal Processing (MMSP)*, IEEE, 2020, pp. 1–6.

[77] A. J. Ben Ali, M. Kouroshli, S. Semenova, Z. S. Hashemifar, S. Y. Ko and K. Dantu, 'Edge-slam: Edge-assisted visual simultaneous localization and mapping,' *ACM Transactions on Embedded Computing Systems*, vol. 22, no. 1, pp. 1–31, 2022.

[78] J. Hofer, P. Sossalla, C. L. Vielhaus, J. Rischke, M. Reisslein and F. H. Fitzek, 'Comparison of analyze-then-compress methods in edge-assisted visual slam,' *IEEE Access*, vol. 11, pp. 68 728–68 743, 2023.

[79] A. J. Hussain, A. Al-Fayadh and N. Radi, 'Image compression techniques: A survey in lossless and lossy algorithms,' *Neurocomputing*, vol. 300, pp. 44–69, 2018.

[80] L. Riazuelo, J. Civera and J. M. Montiel, 'C2tam: A cloud framework for cooperative tracking and mapping,' *Robotics and Autonomous Systems*, vol. 62, no. 4, pp. 401–413, 2014.

[81] G. Mohanarajah, V. Usenko, M. Singh, R. D'Andrea and M. Waibel, 'Cloud-based collaborative 3d mapping in real-time with low-cost robots,' *IEEE Transactions on Automation Science and Engineering*, vol. 12, no. 2, pp. 423–431, 2015.

[82] F. Nenci, L. Spinello and C. Stachniss, 'Effective compression of range data streams for remote robot operations using h. 264,' in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, 2014, pp. 3794–3799.

[83] J. Xu et al., '{Swarmmap}: Scaling up real-time collaborative visual {slam} at the edge,' in *19th USENIX Symposium on Networked Systems Design and Implementation (NSDI 22)*, 2022, pp. 977–993.

[84] D. Mishra, S. K. Singh and R. K. Singh, 'Deep architectures for image compression: A critical review,' *Signal Processing*, vol. 191, p. 108 346, 2022.

[85] S. Jamil, M. J. Piran, M. Rahman and O.-J. Kwon, 'Learning-driven lossy image compression: A comprehensive survey,' *Engineering Applications of Artificial Intelligence*, vol. 123, p. 106 361, 2023.

[86] M. A. Rahman and M. Hamada, 'Lossless image compression techniques: A state-of-the-art survey,' *Symmetry*, vol. 11, no. 10, p. 1274, 2019.

[87] I. A. Urbaniak, 'Using compressed jpeg and jpeg2000 medical images in deep learning: A review,' *Applied Sciences*, vol. 14, no. 22, p. 10 524, 2024.

[88] E. Rublee, V. Rabaud, K. Konolige and G. Bradski, 'Orb: An efficient alternative to sift or surf,' in *2011 International conference on computer vision*, Ieee, 2011, pp. 2564–2571.

[89] H. Bay, A. Ess, T. Tuytelaars and L. Van Gool, 'Speeded-up robust features (surf),' *Computer vision and image understanding*, vol. 110, no. 3, pp. 346–359, 2008.

[90] D. G. Lowe, 'Distinctive image features from scale-invariant keypoints,' *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[91] E. Rosten and T. Drummond, 'Machine learning for high-speed corner detection,' in *European conference on computer vision*, Springer, 2006, pp. 430–443.

[92] S. Shao, 'A monocular slam system based on the orb features,' in *2023 IEEE 3rd International Conference on Power, Electronics and Computer Applications (ICPECA)*, IEEE, 2023, pp. 1221–1231.

[93] Z. Hu et al., 'Optimized feature points and keyframe methods for vslam in high-dynamic indoor environments,' *IEEE Transactions on Intelligent Transportation Systems*, vol. 26, no. 3, pp. 3101–3114, 2025.

[94] W. Fang, Y. Zhang, B. Yu and S. Liu, 'Fpga-based orb feature extraction for real-time visual slam,' in *2017 International Conference on Field Programmable Technology (ICFPT)*, IEEE, 2017, pp. 275–278.

[95] R. Mur-Artal, J. M. M. Montiel and J. D. Tardos, 'Orb-slam: A versatile and accurate monocular slam system,' *IEEE transactions on robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[96] A. R. Sahili et al., 'A survey of visual slam methods,' *IEEE access*, vol. 11, pp. 139 643–139 677, 2023.

[97] R. Mur-Artal and J. D. Tardós, 'Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras,' *IEEE transactions on robotics*, vol. 33, no. 5, pp. 1255–1262, 2017.

[98] J. Sturm, N. Engelhard, F. Endres, W. Burgard and D. Cremers, 'A benchmark for the evaluation of rgb-d slam systems,' in *2012 IEEE/RSJ international conference on intelligent robots and systems*, IEEE, 2012, pp. 573–580.

[99] B. Al-Tawil, T. Hempel, A. Abdelrahman and A. Al-Hamadi, 'A review of visual slam for robotics: Evolution, properties, and future applications,' *Frontiers in Robotics and AI*, vol. 11, p. 1 347 985, 2024.

[100] D. Zou, P. Tan and W. Yu, 'Collaborative visual slam for multiple agents: A brief survey,' *Virtual Reality & Intelligent Hardware*, vol. 1, no. 5, pp. 461–482, 2019.

[101] N. Chebrolu, D. Marquez-Gamez and P. Martinet, 'Collaborative visual slam framework for a multi-robot system,' in *7th Workshop on Planning, Perception and Navigation for Intelligent Vehicles*, 2015.

[102] L. Zheng, K. Xu, J. Jiang, M. Wei, B. Zhou and H. Cheng, 'Real-time efficient environment compression and sharing for multi-robot cooperative systems,' *IEEE Transactions on Intelligent Vehicles*, 2024.

[103] G. S. Martins, D. Portugal and R. P. Rocha, 'A comparison of general-purpose foss compression techniques for efficient communication in cooperative multi-robot tasks,' in *2014 11th international conference on informatics in control, automation and robotics (ICINCO)*, IEEE, vol. 2, 2014, pp. 136–147.

[104] G. S. Martins, D. Portugal and R. P. Rocha, 'On the usage of general-purpose compression techniques for the optimization of inter-robot communication,' in *Informatics in Control, Automation and Robotics: 11th International Conference, ICINCO 2014 Vienna, Austria, September 2-4, 2014 Revised Selected Papers*, Springer, 2015, pp. 223–240.

[105] L. Zhang and J. Deng, 'Deep compressed communication and application in multi-robot 2d-lidar slam: An intelligent huffman algorithm,' *Sensors*, vol. 24, no. 10, p. 3154, 2024.

[106] X. Liu, S. Wen, J. Zhao, T. Z. Qiu and H. Zhang, 'Edge-assisted multi-robot visual-inertial slam with efficient communication,' *IEEE Transactions on Automation Science and Engineering*, vol. 22, pp. 2186–2198, 2024.

[107] R. Zhong, Y. Bai, A. Wang, Z. Hu and H. Fang, 'Centralized multi-robot visual slam and scene reconstruction,' in *2024 43rd Chinese Control Conference (CCC)*, IEEE, 2024, pp. 3748–3754.

[108] N. Tahir and R. Parasuraman, 'Edge computing and its application in robotics: A survey,' *Journal of Sensor and Actuator Networks*, vol. 14, no. 4, p. 65, 2025.

[109] X. Jin et al., 'High throughput vlsi design for real-time jpeg 2000 embedded block coding,' *Journal of Real-Time Image Processing*, vol. 22, no. 1, p. 36, 2025.

[110] J. Liu, G. Lu, Z. Hu and D. Xu, 'A unified end-to-end framework for efficient deep image compression,' *arXiv preprint arXiv:2002.03370*, 2020.

[111] Z. Zhang, S. Esenlik, Y. Wu, M. Wang, K. Zhang and L. Zhang, 'End-to-end learning-based image compression with a decoupled framework,' *IEEE transactions on circuits and systems for video technology*, vol. 34, no. 5, pp. 3067–3081, 2023.

[112] S. Wang, Z. Wang, S. Wang and Y. Ye, 'End-to-end compression towards machine vision: Network architecture design and optimization,' *IEEE Open Journal of Circuits and Systems*, vol. 2, pp. 675–685, 2021.

[113] N. Le, H. Zhang, F. Cricri, R. Ghaznavi-Youvalari and E. Rahtu, 'Image coding for machines: An end-to-end learned approach,' in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2021, pp. 1590–1594.

[114] H. Shao, B. Liu, Z. Li, C. Yan, Y. Sun and T. Wang, 'A high-throughput processor for gdn-based deep learning image compression,' *Electronics*, vol. 12, no. 10, p. 2289, 2023.

[115] H. Sun, Q. Yi, F. Lin, L. Yu, J. Katto and M. Fujita, 'F-lic: Fpga-based learned image compression with a fine-grained pipeline,' in *2022 IEEE Asian Solid-State Circuits Conference (A-SSCC)*, IEEE, 2022, pp. 1–3.

[116] H. Sun, Q. Yi and M. Fujita, 'Fpga codec system of learned image compression with algorithm-architecture co-optimization,' *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 14, no. 2, pp. 334–347, 2024.

[117] M. A. Eddine et al., 'Security and real-time fpga integration for learned image compression,' *IEEE Transactions on Multimedia*, 2026.

[118] X. Li, L. Zhou and Y. Zhu, 'Real-time lossless compression system for bayer pattern images with a modified jpeg-ls,' *Mathematics*, vol. 13, no. 20, p. 3245, 2025.

[119] L. Xie, Z. Zhang and T. Zhao, 'A real-time image row-compression method for high-definition usb cameras based on fpga,' *IEEE Access*, vol. 12, pp. 64 663–64 671, 2024.

[120] Z. Yang, Y. Liang, D. Lin, J. Li, Z. Chen and X. Li, 'Real-time stereo vision hardware accelerator: Fusion of sad and adaptive census algorithm,' *IEEE Access*, vol. 12, pp. 154 975–154 989, 2024.

[121] M. Hirose, S. Imaizumi and H. Kiya, 'Encryption method for jpeg bitstreams for partially disclosing visual information,' *Electronics*, vol. 13, no. 11, p. 2016, 2024.

[122] C. Saranya, K. Ponnusamy, R. Ramya, R. Jagadeesan and D. Poornakumar, 'Performance analysis of fpga-based digital filter architectures for image processing applications,' *Circuits, Systems, and Signal Processing*, pp. 1–24, 2026.

Appendix A
Chapter 2 Supplementary Results



Decompressed size=201 KB Quantization factor =25 | Decompressed size=108.4 KB Quantization factor=60 | Decompressed size=71.4 KB Quantization factor=100

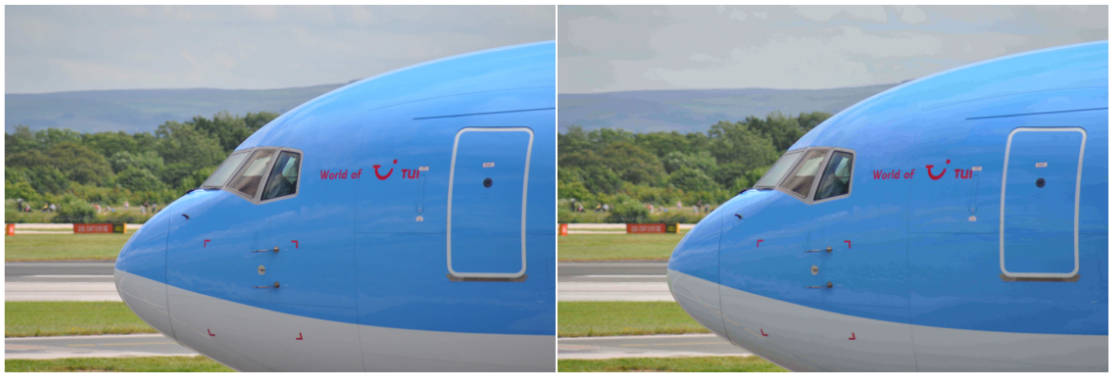
(a) Decompressed Lion image, dimension = 1200 x 1200



Decompressed size=228 KB Quantization factor=1 | Decompressed size=91.7 KB Quantization factor=30 | Decompressed size=47.8 KB Quantization factor=60

(b) Decompressed Apple image, dimension = 1200 x 1200

Figure A.1: (a) and (b) Compression results for the Lion and Apple images using the proposed method with different quantization factors.



Compressed size=437.4 KB
Quantization factor=10

Compressed size=182.8 KB
Quantization factor =25

(a) Decompressed Boeing 777 image, dimension = 1800 x 1196

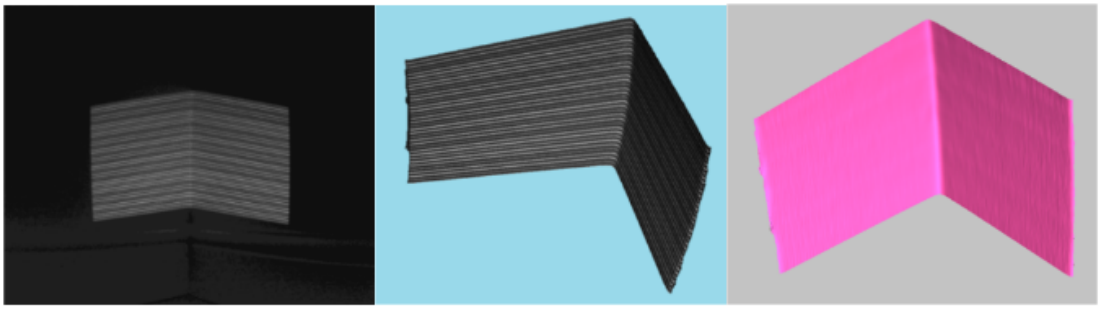


Compressed size=426.3 KB
Quantization factor =25

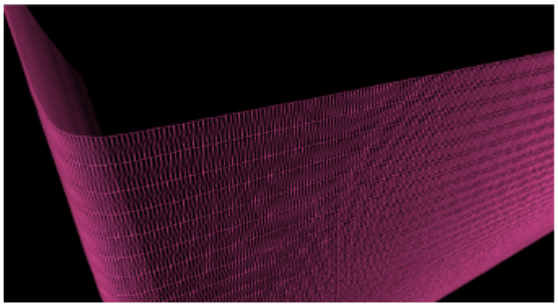
Compressed size=309.8 KB
Quantization factor =35

(b) Decompressed Baghdad image, dimension = 2000 x 1500

Figure A.2: (a) and (b): Boeing 777 and Baghdad color images are compressed by the proposed method using different quantization factors.

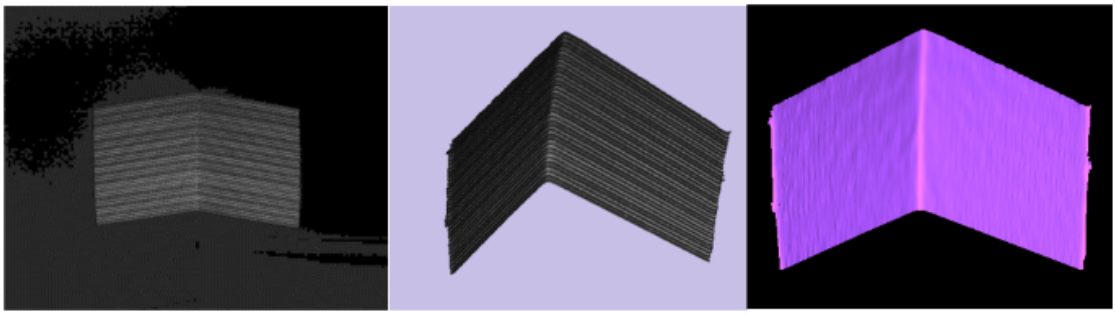


(left) Decompressed 2D Corner image, (middle and right) converted to 3D surface by GMPR software.

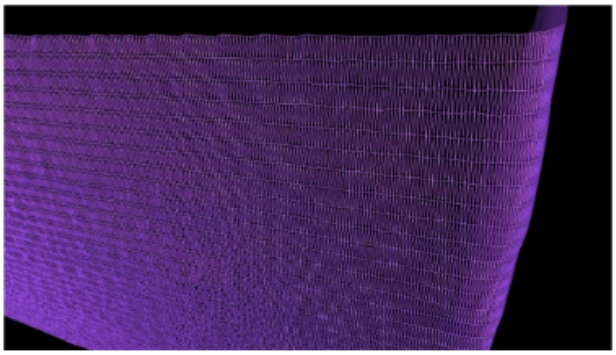


Zooming in the decompressed 2D Corner image showing the mesh details.

(a) Decompressed Corner image, RMSE=4.7 and compressed image size =35.4 KB.



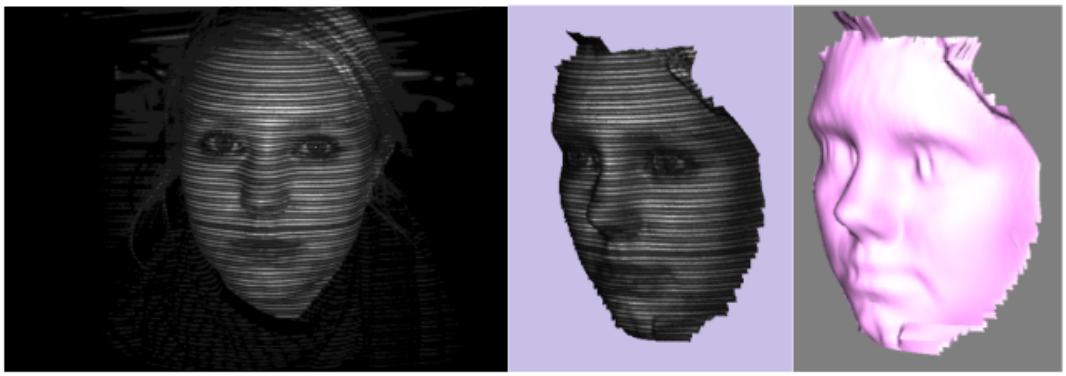
(left) Decompressed 2D Corner image, (middle and right) converted to 3D surface by GMPR software.



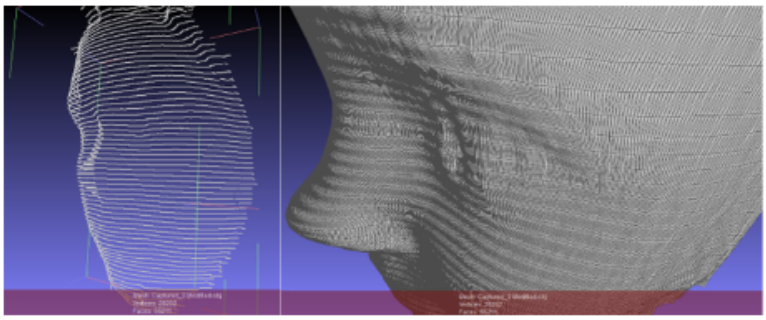
Zooming in the decompressed 2D Corner image showing the mesh details.

(b) Decompressed Corner image, RMSE=15.5 and compressed image size=17.8 KB.

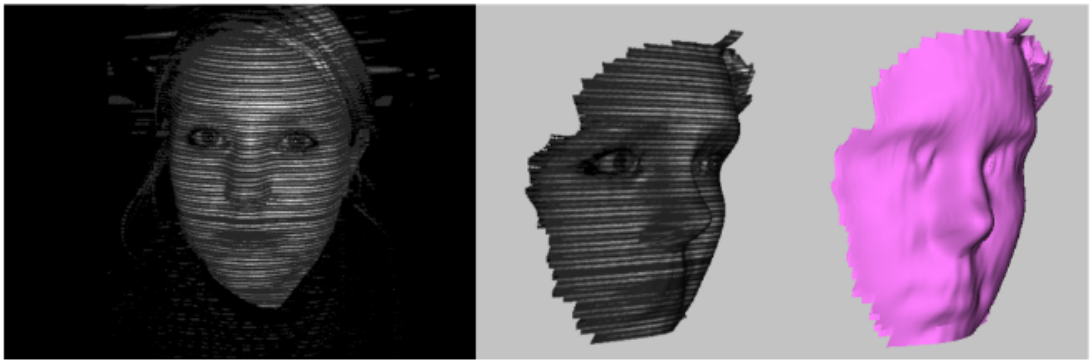
Figure A.3: (a) and (b) Decompressed 2D Corner image used for 3D reconstruction. The 3D mesh surface, consisting vertices and triangles, is successfully reconstructed without significant distortion at high compression ratios of up to 98.6%.



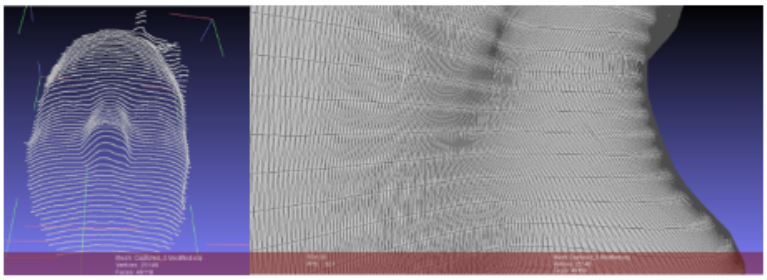
(left) Decompressed 2D Face-1 image, (middle and right) converted to 3D surface using GMPR software.



Reconstructed vertices and mesh successfully without distortion (3D surface showed in MeshLab software).
(a) Decompressed 2D Face-1 image, RMSE=8.4 and compressed size=34KB.

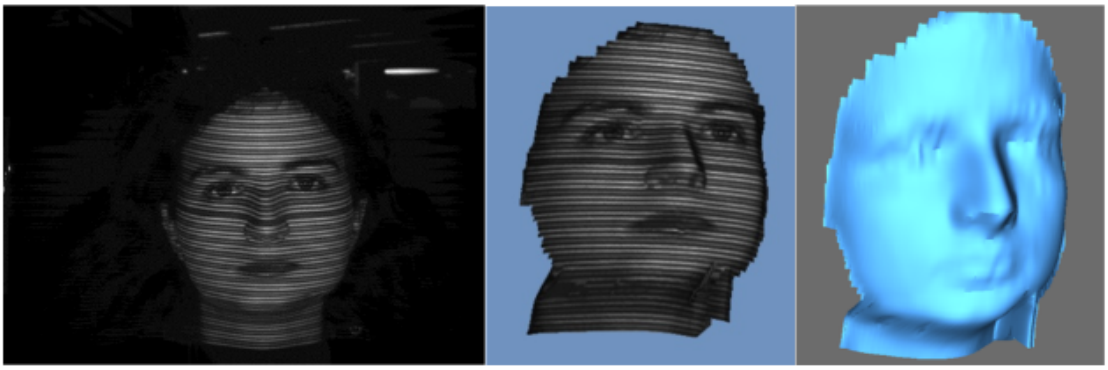


(left) Decompressed 2D Face-1 image, (middle and right) converted to 3D surface using GMPR software.

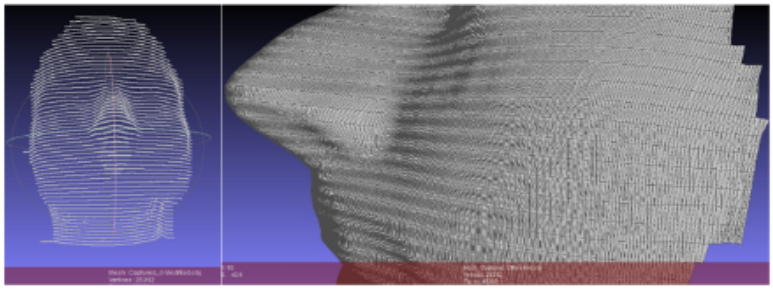


Reconstructed vertices and mesh successfully without distortion (3D surface showed in MeshLab software).
(b) Decompressed 2D Face-1 image, RMSE=11.5 and compressed size =18.1 KB.

Figure A.4: (a) and (b) Decompressed 2D Face-1 image used for 3D reconstruction. The 3D mesh surface, consisting vertices and triangles, is successfully reconstructed without significant distortion at high compression ratios of 98.6%.

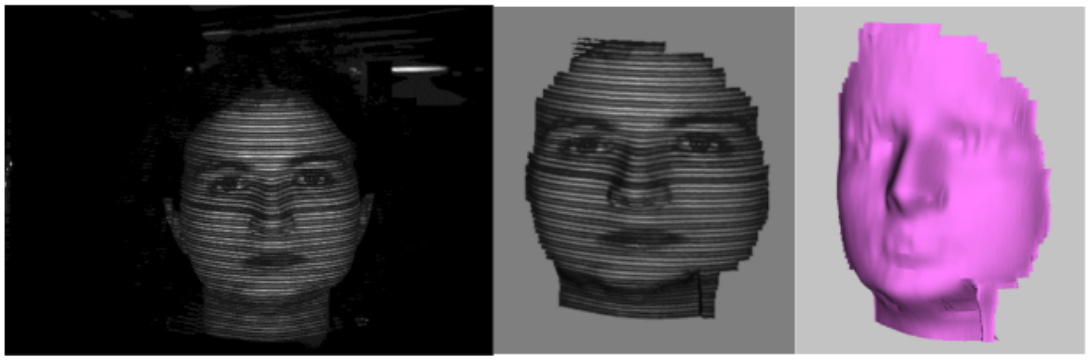


(left) Decompressed 2D Face-2 image, (middle and right) converted to 3D surface using GMPR software.

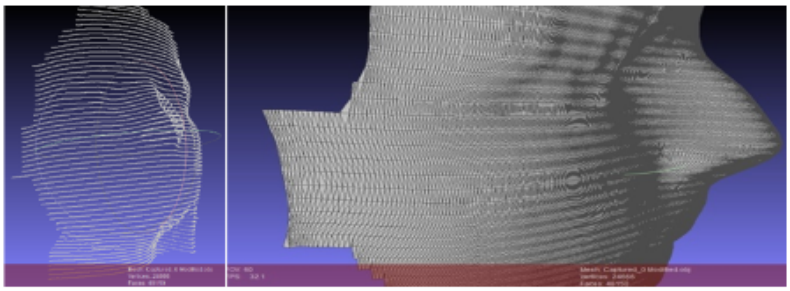


Reconstructed vertices and mesh successfully without distortion (3D surface showed in MeshLab software).

(a) Decompressed 2D Face-2 image, RMSE=6.7 and compressed size=46.2KB.



(left) Decompressed 2D Face-2 image, (middle and right) converted to 3D surface using GMPR software.



Reconstructed vertices and mesh successfully without distortion (3D surface showed in MeshLab software).

(b) Decompressed 2D Face-2 image, RMSE=9.9 and compressed size=20.1KB.

Figure A.5: (a) and (b) Decompressed 2D Face-2 2D image used for 3D reconstruction. 3D mesh surface. The 3D mesh surface, consisting vertices and triangles, is successfully reconstructed without significant distortion at high compression ratios of 98.5%.

Table A.1: Comparison between the DFT and the proposed method (DFT with the matrix reduction algorithm) in terms of image quality and compressed size.

Image	**Size [MB]**	**QF**	**DFT alone**		**RMSE**	**PSNR [dB]**	**Proposed**		**RMSE**	**PSNR [dB]**
			[KB]	**bpp**			**[KB]**	**bpp**		
Conventional images										
Lion	1.37	100	808	0.57	3.59	42.5	71	0.050	8.1	39.0
Apples	1.37	60	576	0.41	2.3	44.5	47	0.033	4.6	41.5
Boeing 777	6.15	25	2200	0.35	1.8	45.5	182	0.028	3.9	42.2
Baghdad	8.58	35	3500	0.40	2.3	44.5	309	0.035	5.6	40.6
Structured-light images										
Corner	1.25	100	615	0.48	2.9	43.5	17	0.013	15.5	36.2
Face-1	1.37	160	624	0.44	4.8	41.3	18	0.012	11.5	37.5
Face-2	1.37	150	508	0.36	4.1	42.0	20	0.014	9.9	38.1

Table A.2: Comparison between JPEG and the proposed method in terms of image quality and compressed size.

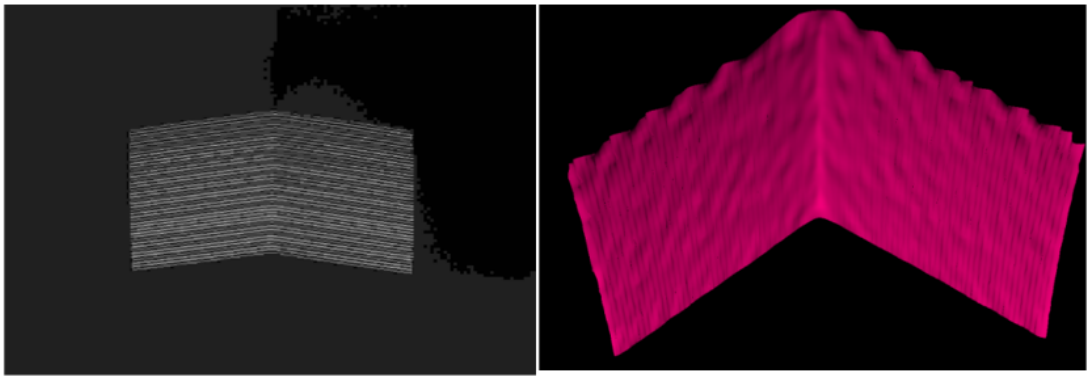
Image	**Size [MB]**	**JPEG**		**RMSE**	**PSNR [dB]**	**Proposed**		**RMSE**	**PSNR [dB]**
		[KB]	**bpp**			**[KB]**	**bpp**		
Conventional images									
Lion	1.37	56	0.039	8.8	38.6	71	0.050	8.1	39.0
Apples	1.37	48	0.034	3.2	43.0	47	0.033	4.6	41.5
Boeing 777	6.15	210	0.033	8.7	38.7	182	0.028	3.9	42.2
Baghdad	8.58	279	0.031	3.5	42.6	309	0.035	5.6	40.6
Structured-light images									
Corner	1.25	26	0.020	14.3	36.5	17	0.013	15.5	36.2
Face-1	1.37	23	0.016	16.5	35.9	18	0.012	11.5	37.5
Face-2	1.37	27	0.019	13.1	36.9	20	0.014	9.9	38.1

Table A.3: Comparison between the conventional matrix reduction approach [39] and the proposed approach in terms of execution time.

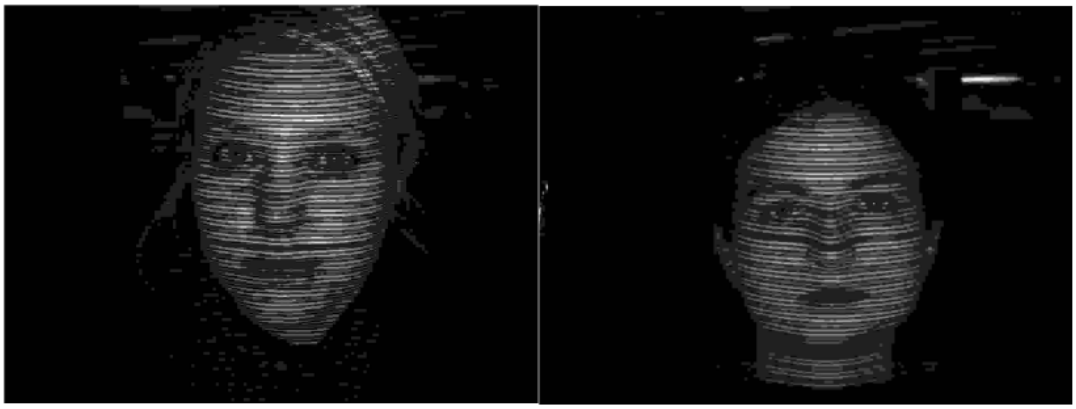
Image	**Size [MB]**	**Conventional matrix reduction**			**Proposed algorithm**		
		[KB]	**bpp**	**Time [s]**	**[KB]**	**bpp**	**Time [s]**
Lion	1.37	98	0.069	240	71	0.050	40
Apples	1.37	92	0.065	90	47	0.033	15
Boeing 777	6.15	240	0.038	420	182	0.028	150
Baghdad	8.58	673	0.076	720	309	0.035	198
Corner	1.25	56	0.043	84	17	0.013	22
Face-1	1.37	46	0.032	144	18	0.012	59
Face-2	1.37	38	0.027	174	20	0.014	66

Table A.4: Comparison between the method in [38] and the proposed approach in terms of image quality and compressed size.

Image	**Size [MB]**	**Conventional matrix reduction [38]**				**Proposed method**			
		Size [KB]	**bpp**	**RMSE**	**PSNR [dB]**	**Size [KB]**	**bpp**	**RMSE**	**PSNR [dB]**
Lion	1.37	98	0.069	10.1	38.0	71	0.050	8.1	39.0
Apples	1.37	92	0.065	7.1	39.6	47	0.033	4.6	41.5
Boeing 777	6.15	240	0.038	10.2	38.0	182	0.028	3.9	42.2
Baghdad	8.58	673	0.076	5.9	40.4	309	0.035	5.6	40.6
Corner	1.25	56	0.043	16.0	36.0	17	0.013	15.5	36.2
Face-1	1.37	46	0.032	14.4	36.5	18	0.012	11.5	37.5
Face-2	1.37	38	0.027	11.2	37.6	20	0.014	9.9	38.1



(a) 3D surface reconstructed from the Corner image, exhibiting some degradation.



(b) The compressed sizes of Face-1 and Face-2 are 23 KB and 27 KB, respectively. The 3D meshes were not reconstructed from these two images because the stripe patterns were not sufficiently distinguishable. Consequently, the system was unable to detect and interpret the stripe lines required for 3D reconstruction.

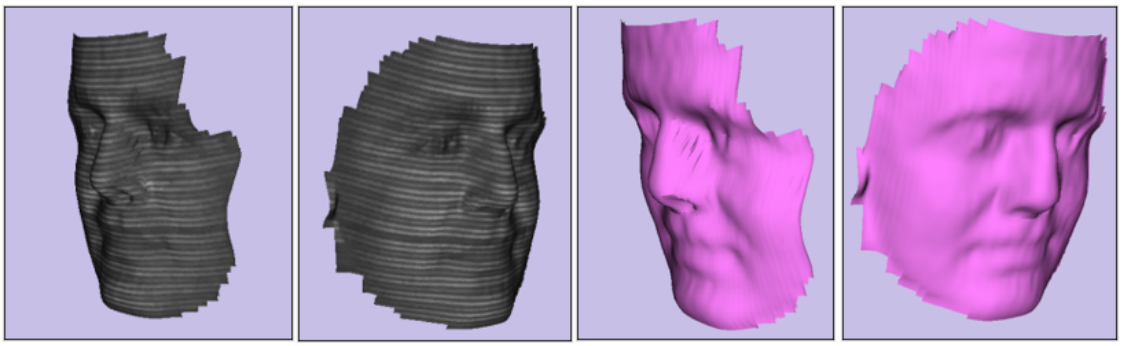
Figure A.6: 3D reconstruction from JPEG compressed images. In (a), reconstruction was possible but but exhibited significant artifacts. In (b), 3D reconstruction was not possible because the images were too degraded.

Appendix B
Chapter 3 Supplementary Results

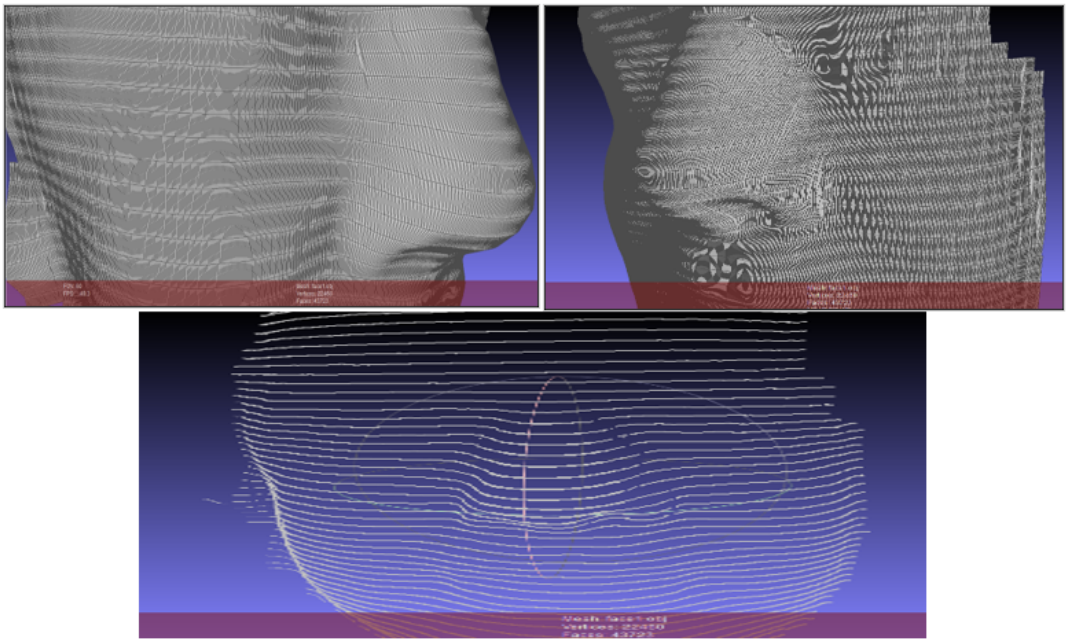


(a) Original image.

(b) Decompressed image by the proposed algorithm.



(c) The 3D object for the decompressed image created by GMPR software.



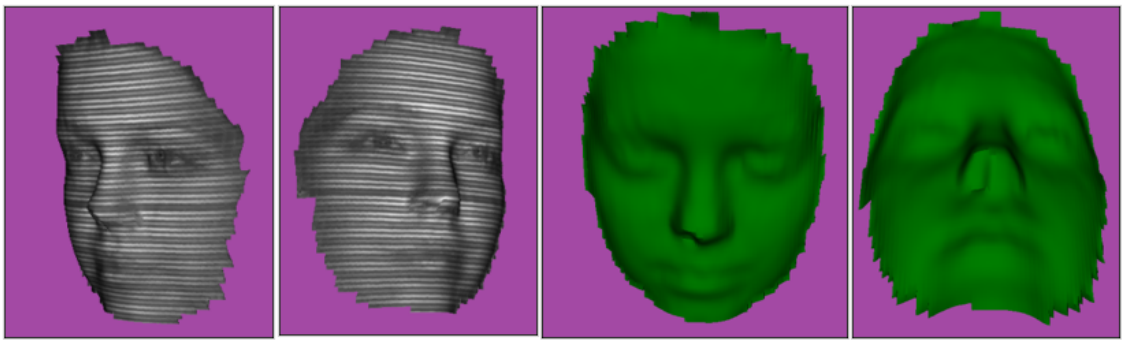
(d) MeshLab software shows 3D mesh surface details created successfully.

Figure B.1: (a, b, c, and d) Decompressed "Face1" image converted successfully to a 3D surface at higher compression ratio (99%).

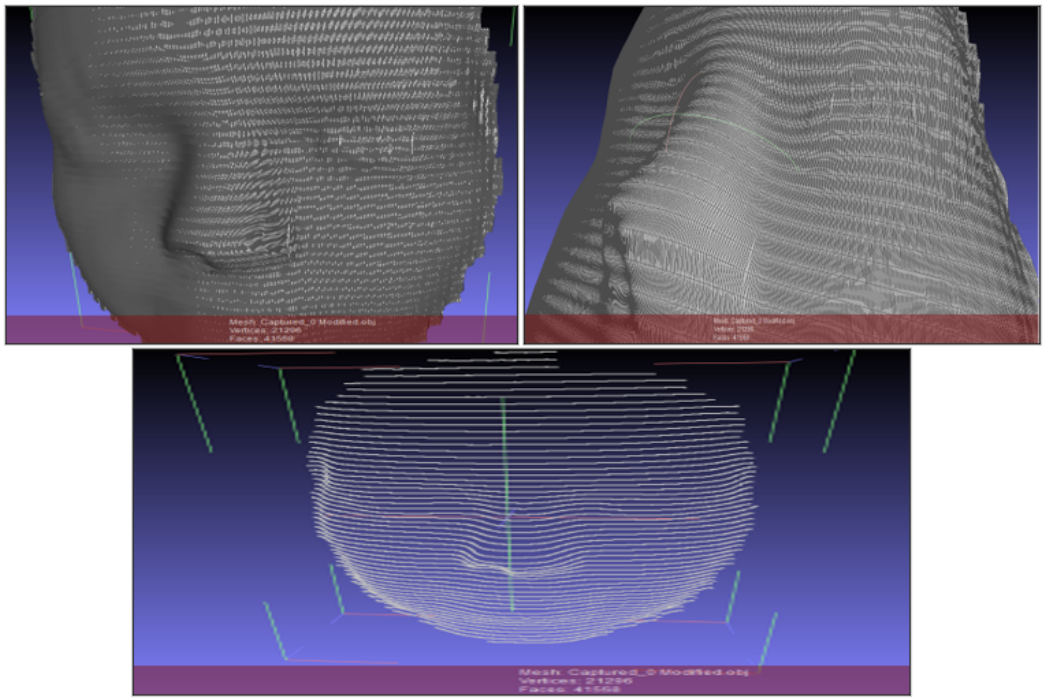


(a) Original image.

(b) Decompressed image by the proposed algorithm.



(c) The 3D object for the decompressed image created by GMPR software.



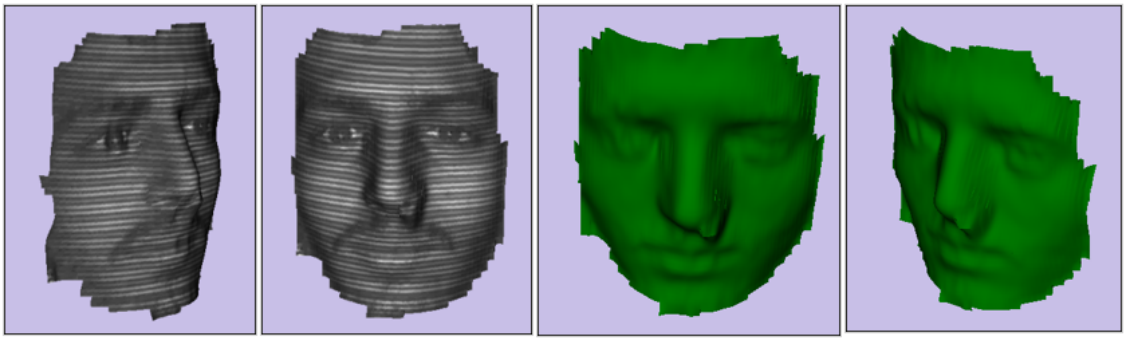
(d) MeshLab software shows 3D mesh surface details created successfully.

Figure B.2: (a, b, c, and d) Decompressed "Face2" image converted successfully to a 3D surface at higher compression ratio (over 98%).

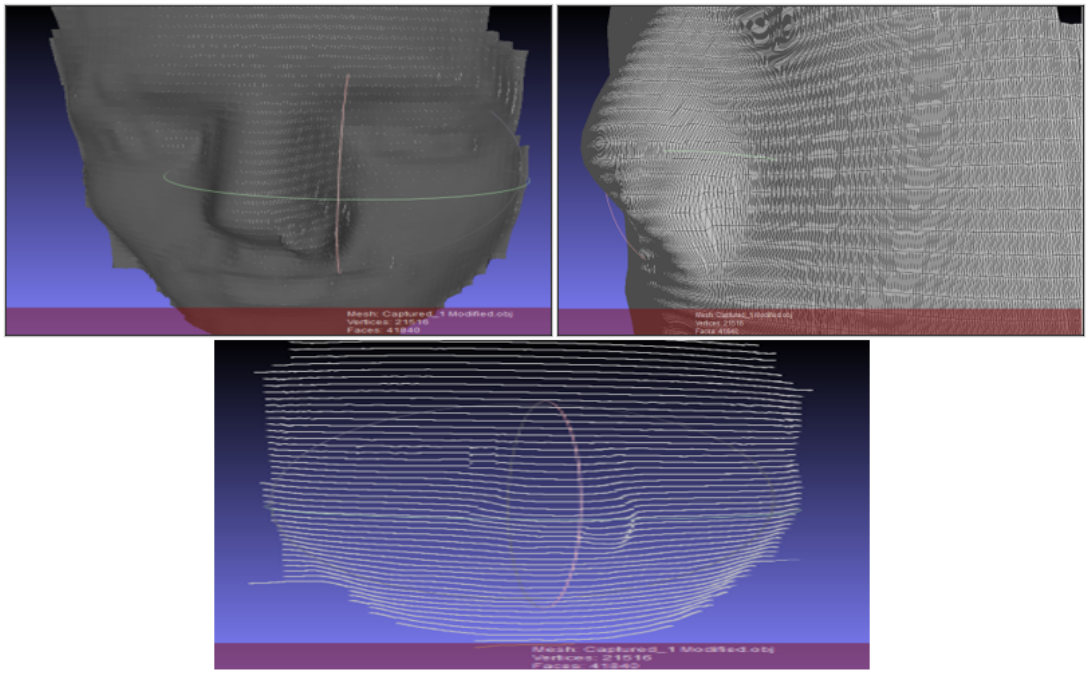


(a) Original image.

(b) Decompressed image by the proposed algorithm.



(c) The 3D object for the decompressed image created by GMPR software.



(d) MeshLab software shows 3D mesh surface details created successfully.

Figure B.3: (a, b, c, and d) Decompressed "Face3" image converted successfully to a 3D surface at higher compression ratio (over 98%).

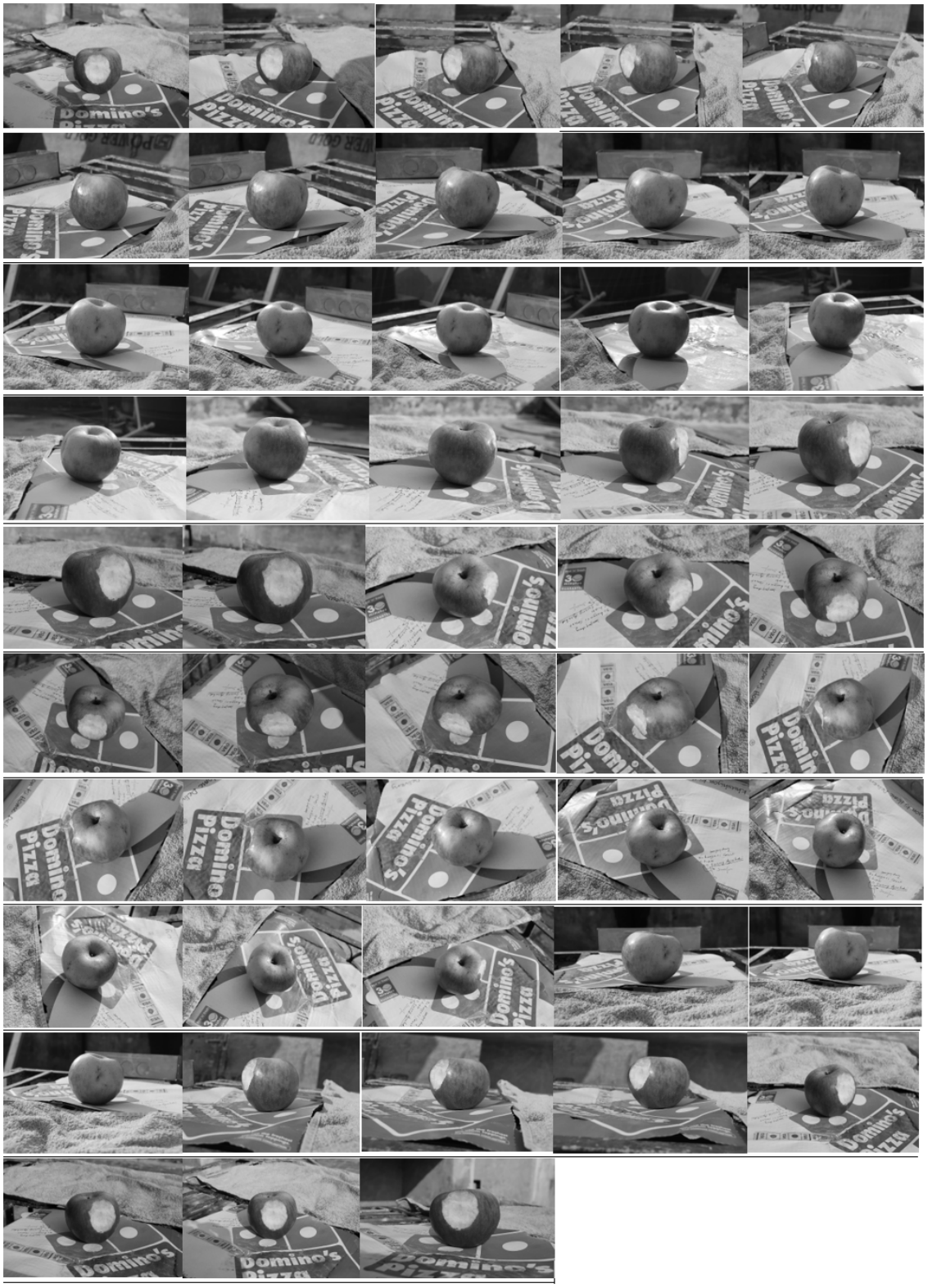
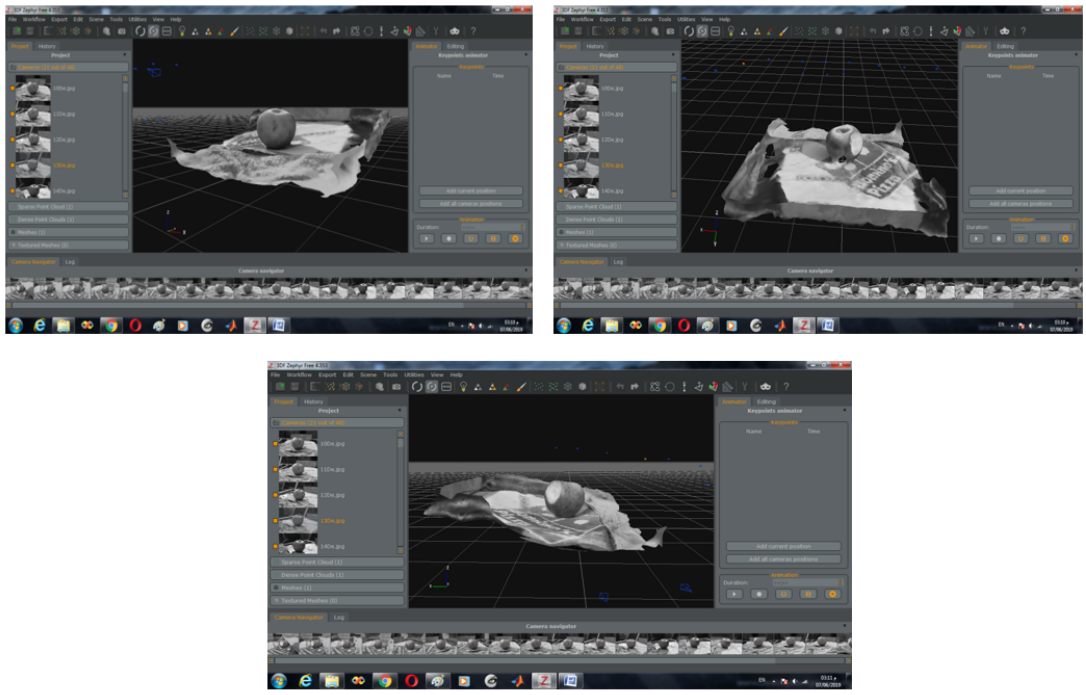
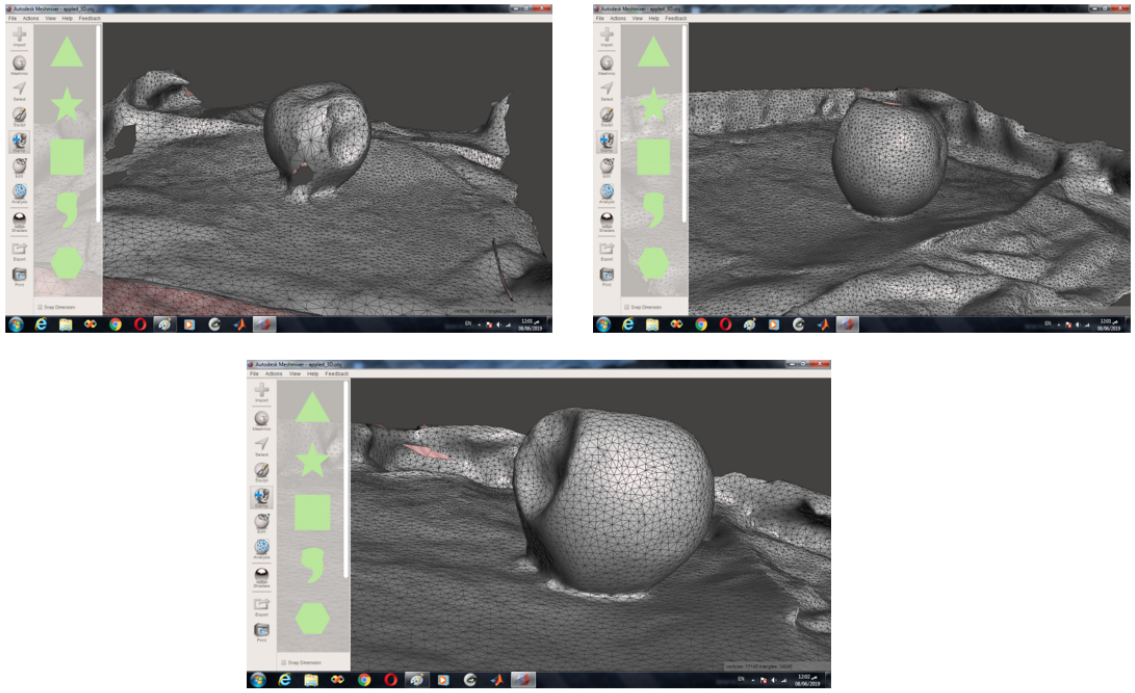


Figure B.4: Decompressed 48 frames of Apple images, dimensions for each image = 1936×1296.



(a) Apple frames converted to 3D object by using 3DF Zephyr Software

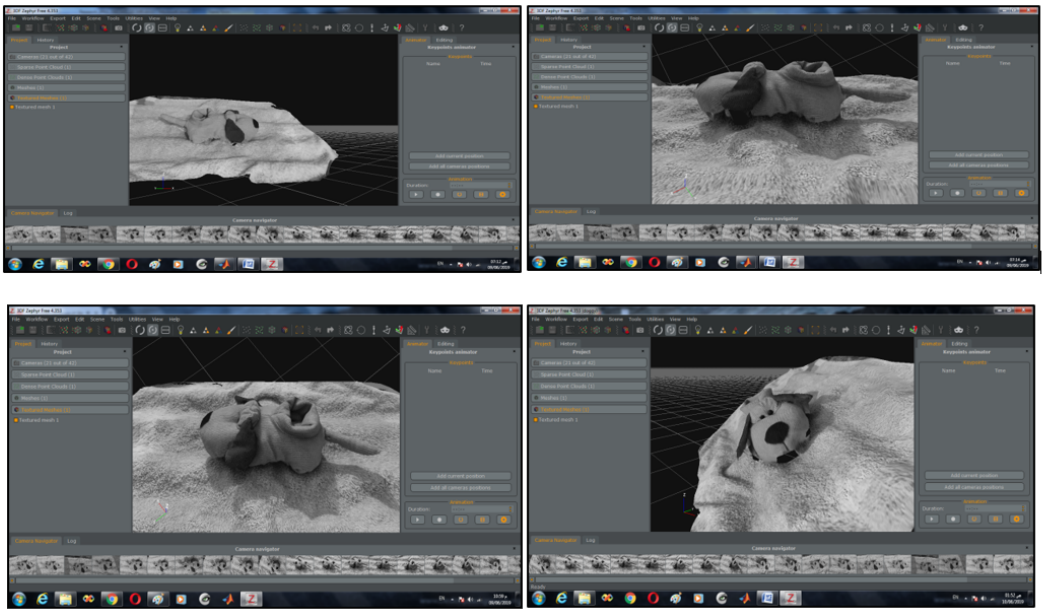


(b) 3D Mesh for the 3D apple object shown by 3D Mesh Mixer software

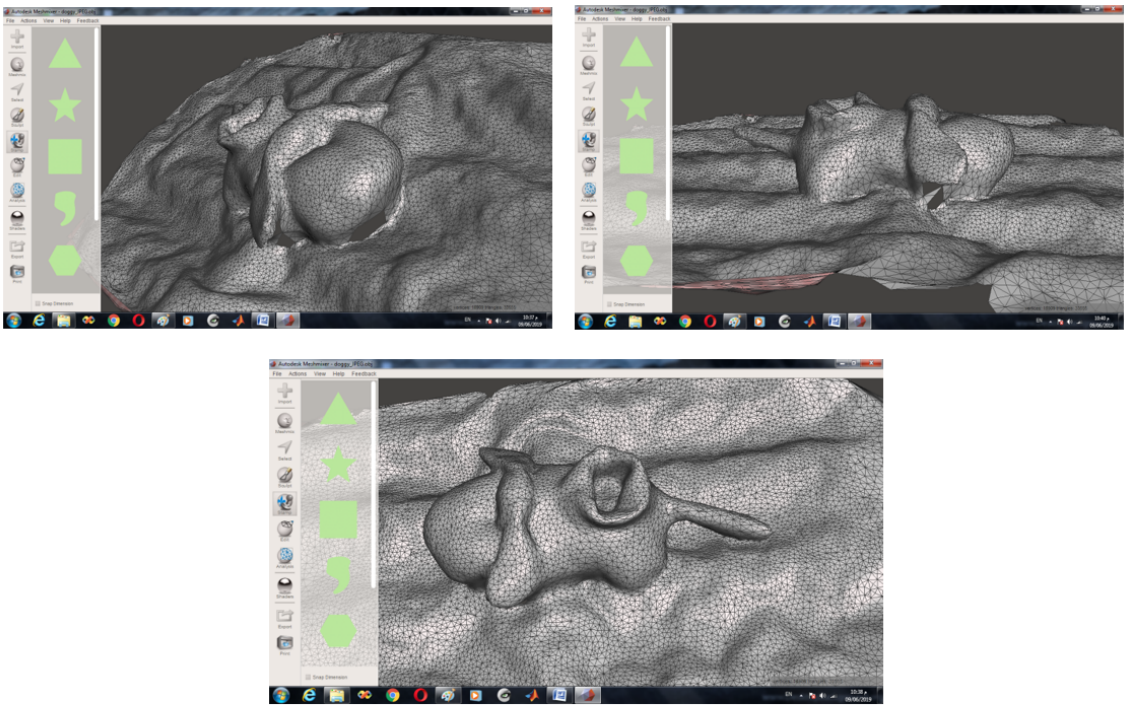
Figure B.5: (a) and (b) show the 3D object and mesh surface for the decompressed Apple images by the proposed compression method at higher compression ratio (97.2%).



Figure B.6: Decompressed 42 frames of Soft Toy images, dimensions for each image = 1936×1296.



(a) Doggy's frames converted to 3D object by using 3DF Zephyr Software

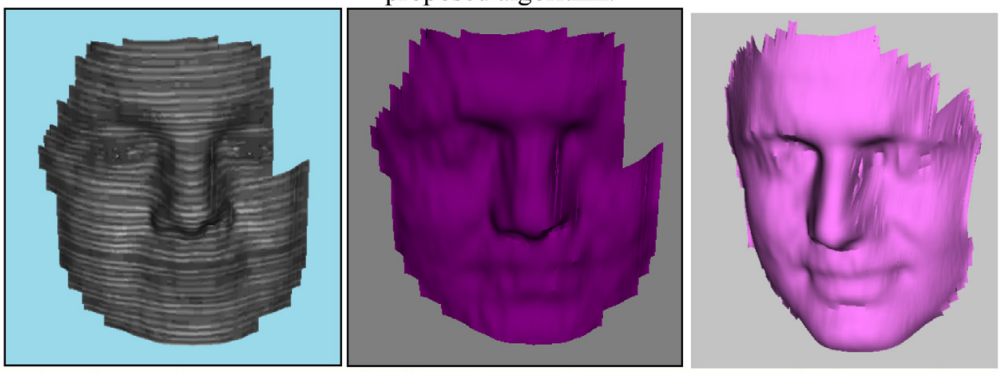


(b) The 3D Mesh for 3D Doggy object showed by 3D Mesh Mixer software

Figure B.7: (a) and (b) show the 3D object and the mesh surface for the compressed Soft Toy images by the proposed method at higher compression ratio (97.6%).



(a) Left - Decompressed image by traditional JPEG technique, (right) Decompressed image by our proposed algorithm.

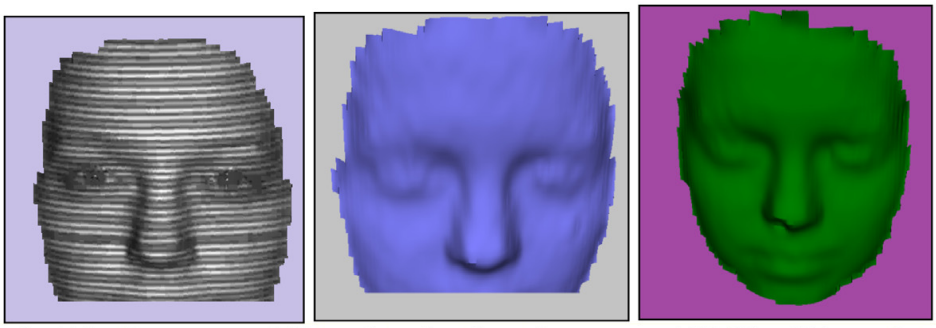


(b) (b) Left and middle - Reconstructed 3D mesh surface from decompressed 2D "Face1" image by traditional JPEG technique, (Right) - Reconstructed 3D mesh surface from decompressed 2D "Face1" image by our proposed algorithm.

Figure B.8: (a) and (b) 3D mesh surface reconstruction for "Face1" comparison of traditional JPEG standard versus proposed method.



(a) Left - Decompressed image by traditional JPEG technique, (right) Decompressed image by our proposed algorithm.

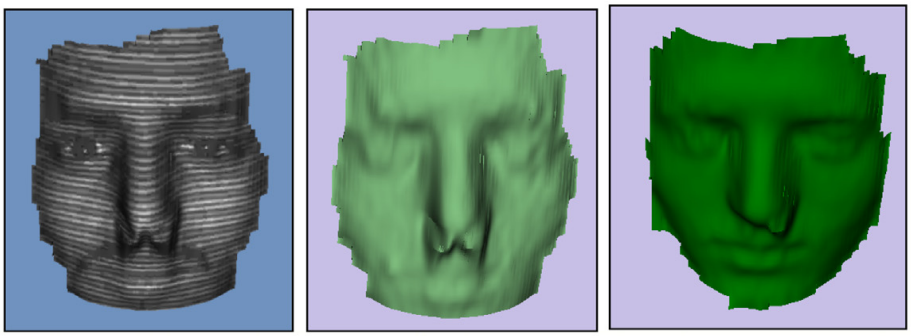


(b) Left and middle- Reconstructed 3D mesh surface from decompressed 2D "Face2" image by traditional JPEG technique, (Right) Reconstructed 3D mesh surface from decompressed 2D "Face2" image by our proposed algorithm.

Figure B.9: (a) and (b) 3D mesh surface reconstruction for "Face2" comparison of traditional JPEG standard versus proposed method.



(a) left - Decompressed image by traditional JPEG technique, (right) Decompressed image by our proposed algorithm.



(b) Left and middle - "Face3" reconstructed 3D mesh surface from decompressed 2D image by traditional JPEG technique, (Right) Reconstructed 3D mesh surface from decompressed 2D "Face2" image by our proposed algorithm.

Figure B.10: (a) and (b) 3D mesh surface reconstruction for "Face3" comparison of traditional JPEG standard versus proposed method.

Table B.1: Results of the conventional JPEG standard and OBJ file sizes for grayscale structured-light images compared with the proposed algorithm.

Frame	**Proposed**				**JPEG standard**			
	Size [KB]	**RMSE**	**PSNR**	**3D .OBJ file size [MB]**	**Size [KB]**	**RMSE**	**PSNR [dB]**	**3D .OBJ file size [MB]**
Face1	14.0	3.3	42.9	6.77	22.0	7.0	39.6	8.42
Face2	17.9	4.35	41.7	6.60	32.1	7.96	39.1	8.16
Face3	21.1	8.3	38.9	6.67	37.8	10.6	37.8	8.82

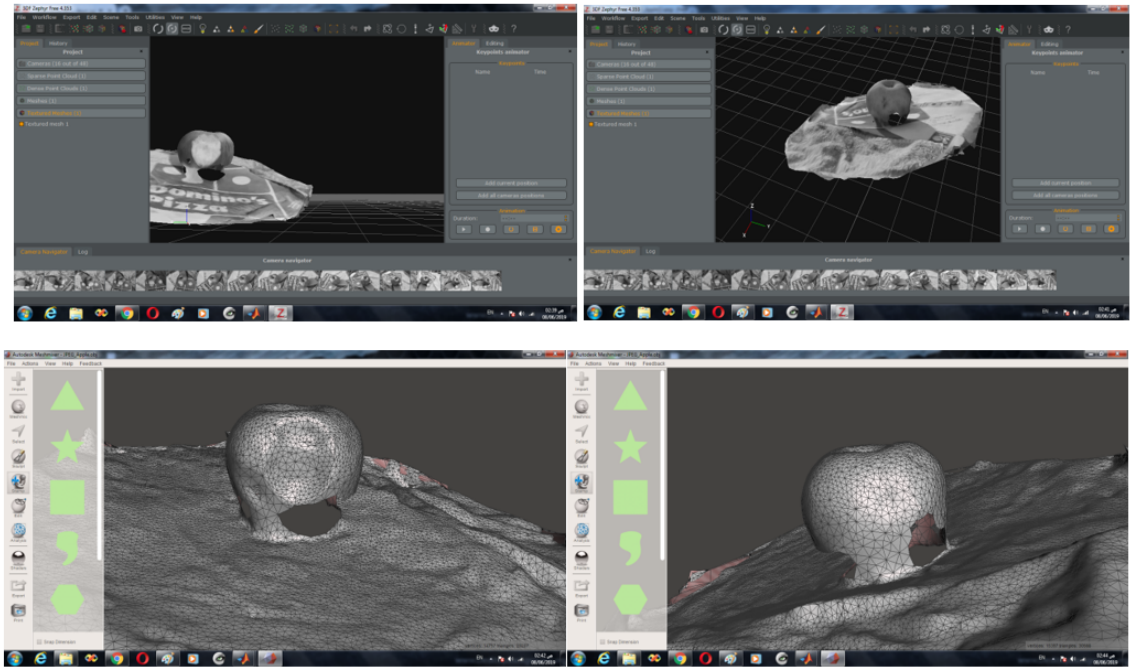


Figure B.11: (top and middle) 3D object and 3D mesh for "Apple" image decompressed by the traditional JPEG algorithm at higher compression ratio of 94%, (bottom) 3D mesh reconstruction at higher compression ratio of 97.2% by the proposed method.

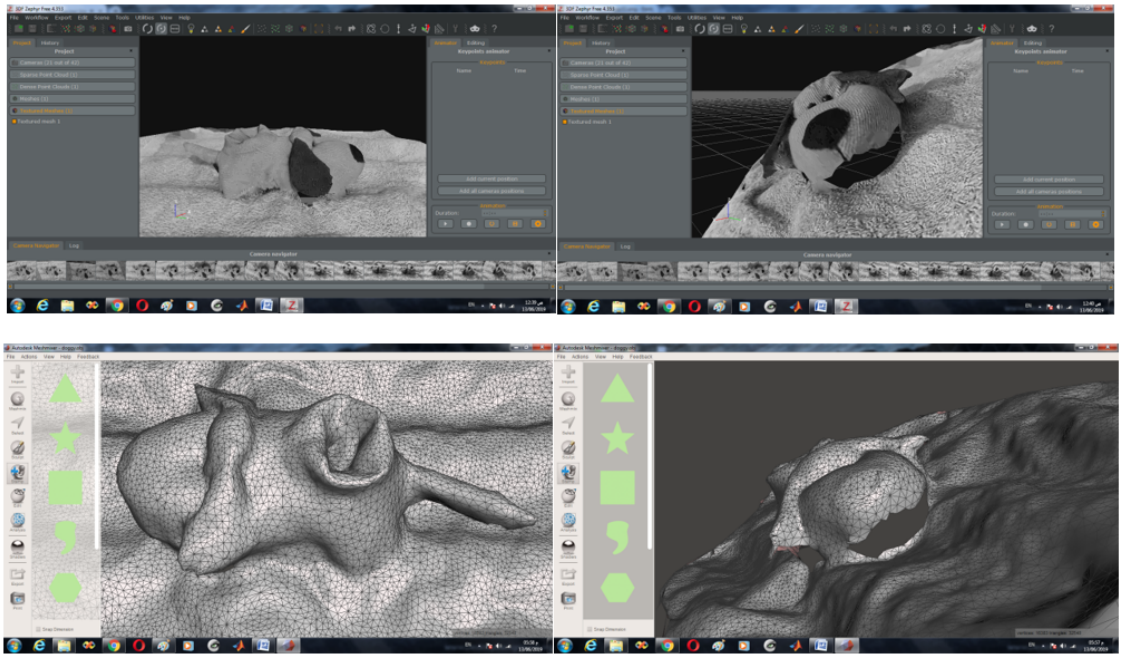


Figure B.12: (top and middle) 3D object and 3D mesh for "Soft Toy" image decompressed by the traditional JPEG algorithm at higher compression ratio of 97%, (bottom) 3D mesh reconstruction at higher compression ratio of 97.6% by the proposed method.

Table B.2: Results of the conventional JPEG standard and OBJ file sizes for multiple-viewpoint images compared with the proposed algorithm.

Sequence	**Proposed**				**JPEG standard**			
	Total size [MB]	**Average RMSE**	**Average PSNR[dB]**	**3D .OBJ file size [MB]**	**Size [MB]**	**Average RMSE**	**Average PSNR [dB]**	**3D .OBJ file size [MB]**
Apple	3.10	5.8	41.3	15.65	4.77	7.4	39.4	19.21
Soft Toy	2.39	11.0	38.2	28.80	5.59	5.7	40.5	44.93

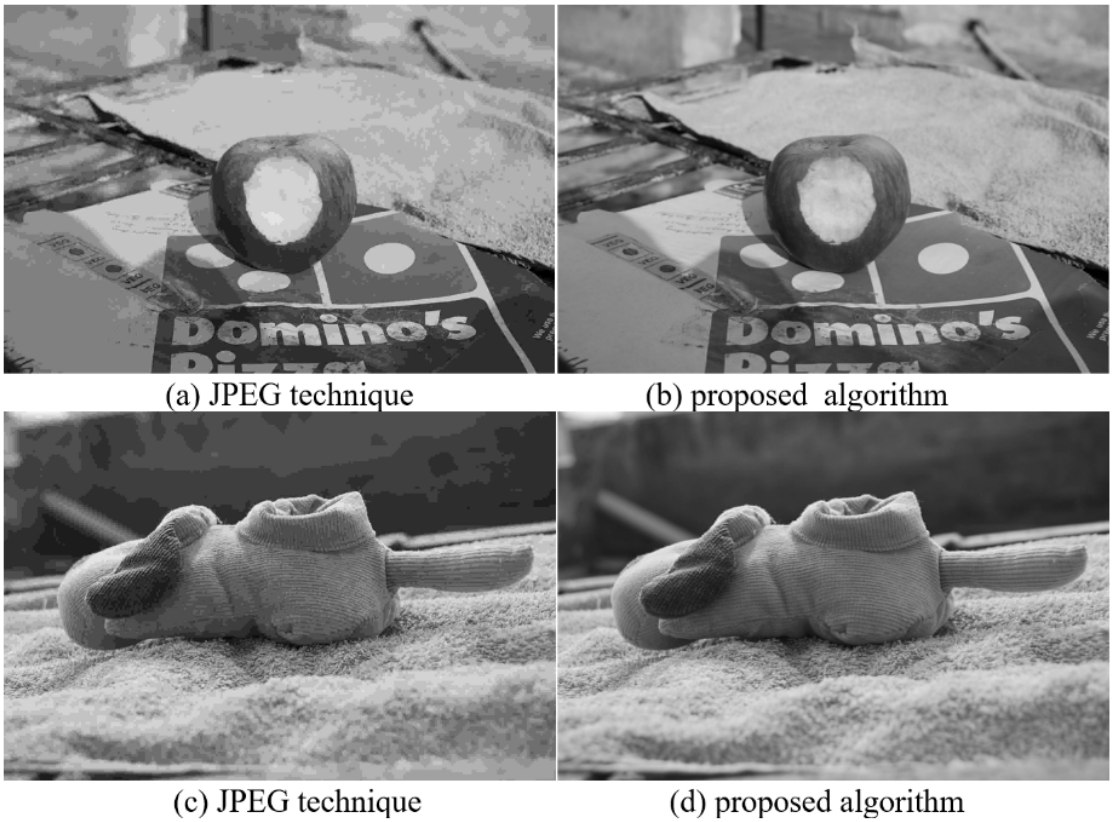


Figure B.13: (a and c) Perceptual assessment: the decompressed image quality shows artifacts at higher compression ratios by the JPEG algorithm, while the other decompressed images (b and d) kept the quality at higher compression ratios by the proposed algorithm.

Appendix C
Chapter 4 Supplementary Results

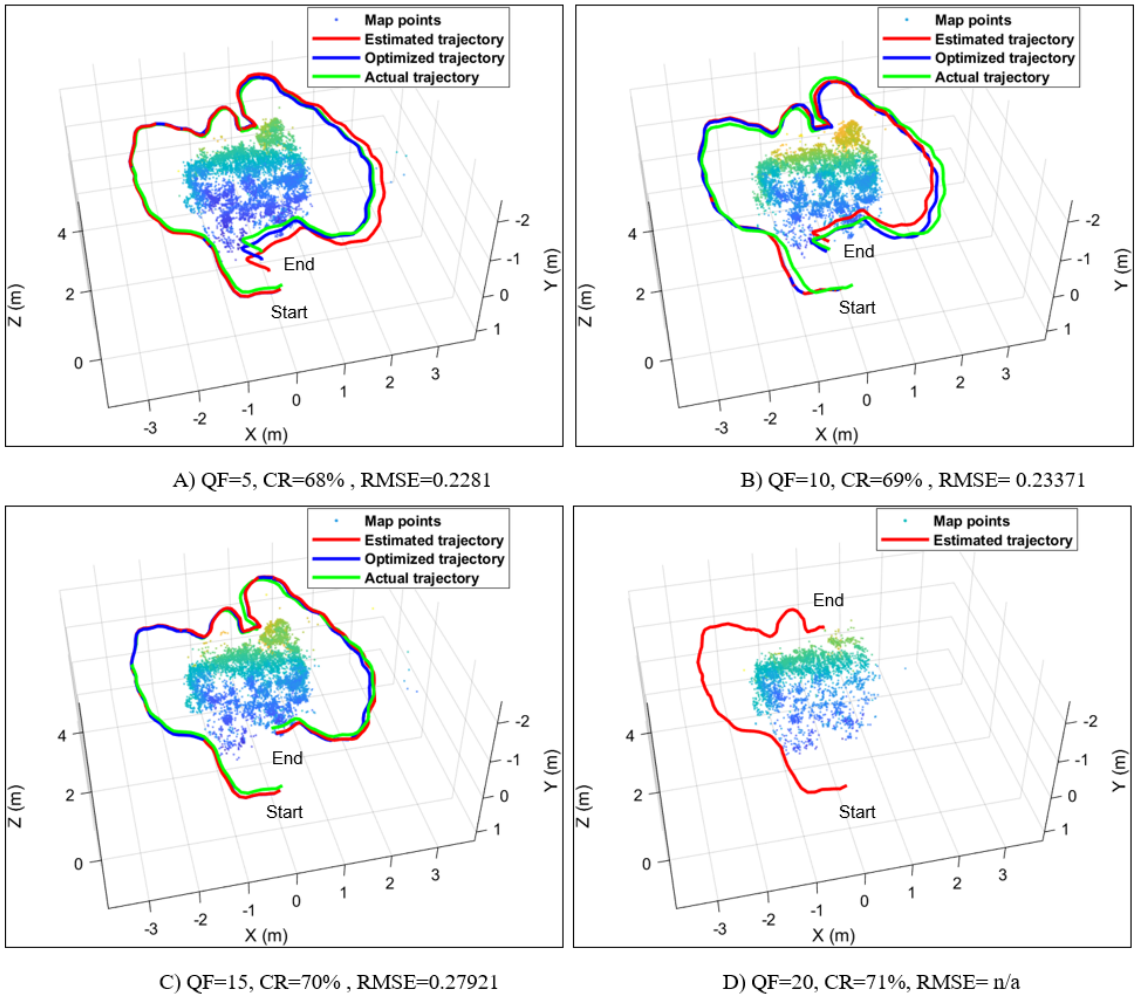


Figure C.1: ORB-SLAM trajectory estimation performance adopting JPEG compression.

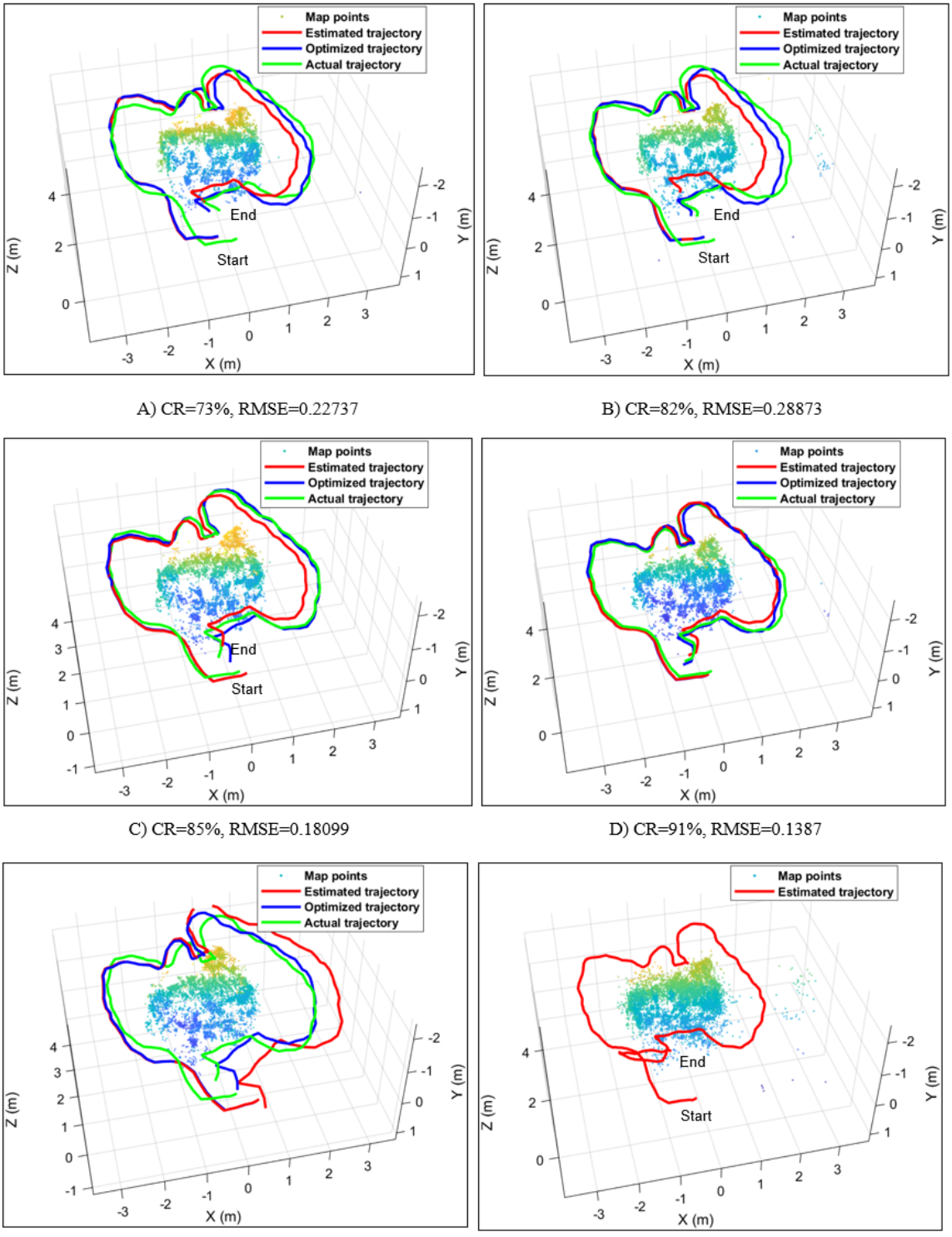
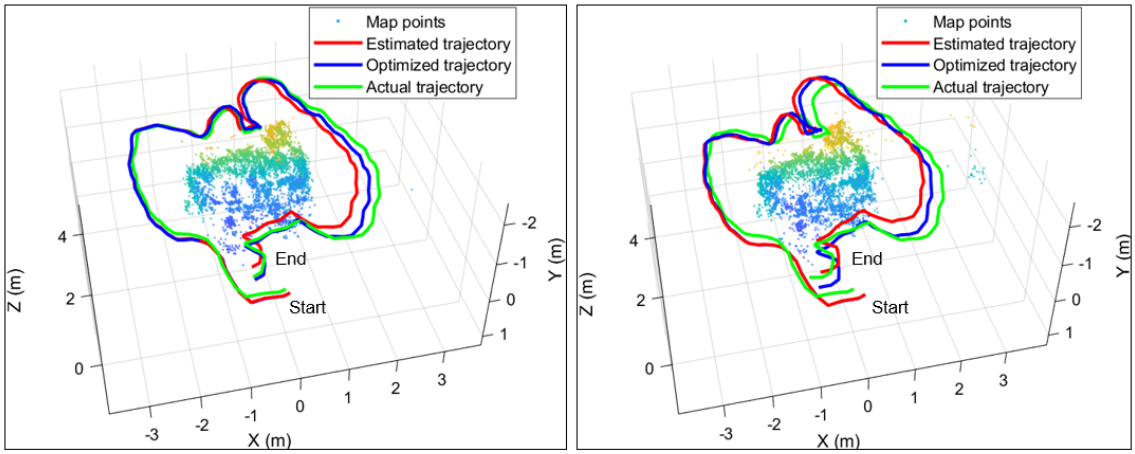
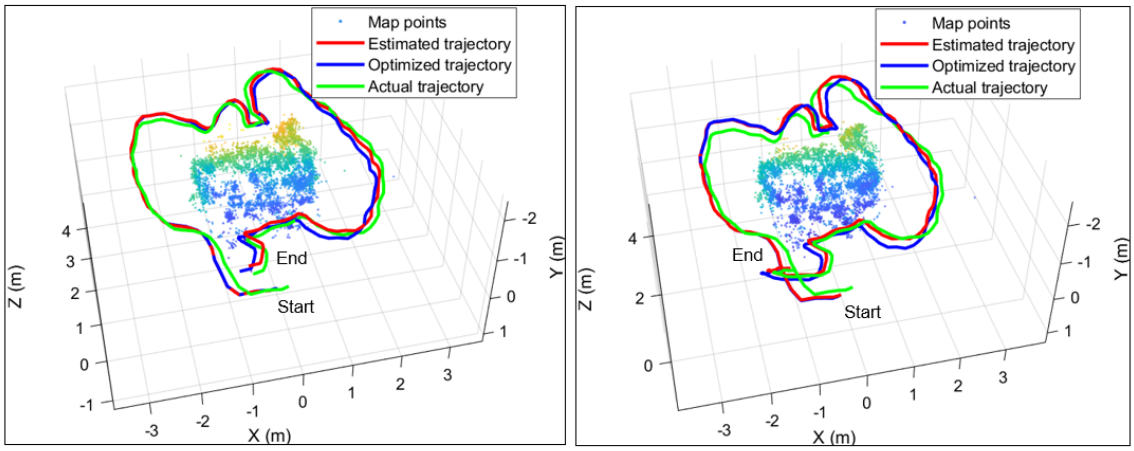


Figure C.2: Trajectory estimation performance of ORB-SLAM system adopting JPEG 2000.



A) QF=5, CR=82%, RMSE=0.13737

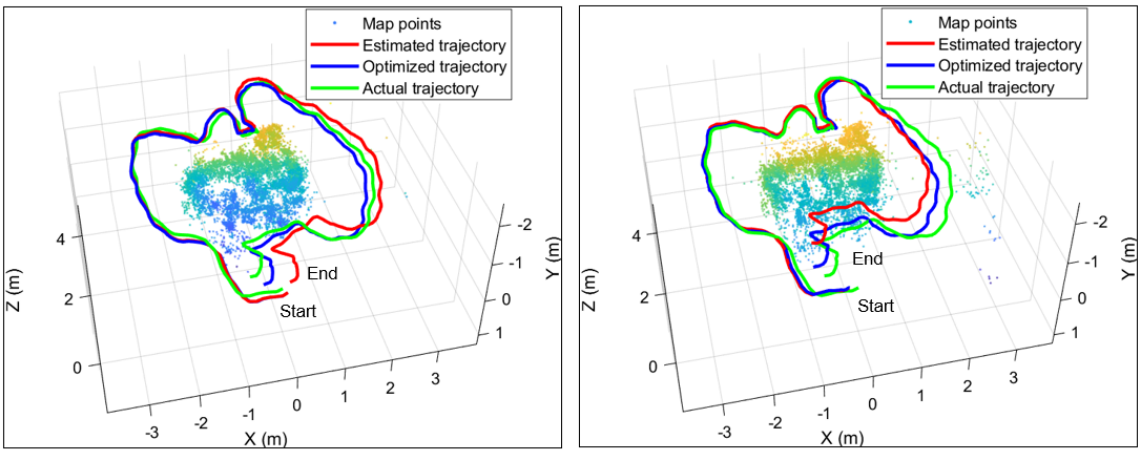
B) QF=10, CR=86%, RMSE=0.2887



C) QF=15, CR=88%, RMSE=0.24905

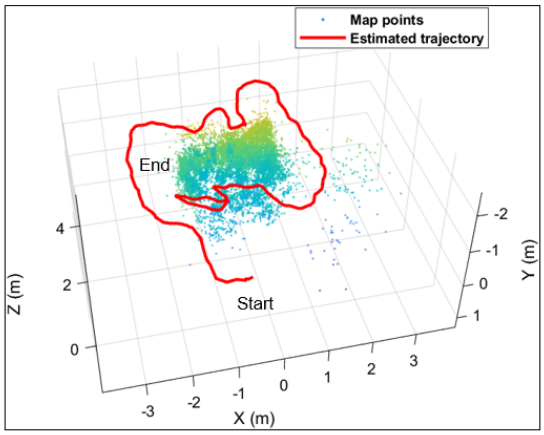
D) QF=20, CR=89%, RMSE=0.3185

Figure C.3: Multi-level based-ORB-SLAM trajectory estimation performance at lower QFs and CRs.



A) QF=50, CR=92%, RMSE= 0.3482

B) QF=75, CR=94%, RMSE=0.3602



C) QF=100, CR=96%, RMSE=n/a

Figure C.4: Multi-level based-ORB-SLAM trajectory estimation performance at higher QFs and CRs.

Table C.1: Encoding performance metrics of the ORB-SLAM integrating the proposed multi-level method.

QF	**PSNR**	**SSIM**	**Single-level implementation**				**Two-level implementation**			
	[dB]		**Size**	**CR**	**Time [s]**		**Size**	**CR**	**Time [s]**	
			[KB]	**[%]**	**Enc.**	**Dec.**	**[KB]**	**[%]**	**Enc.**	**Dec.**
Frame sequence 1132, Frame size 472 KB										
5	35.2333	0.9515	87	82	0.7773	0.7201	86	82	0.8800	0.8307
10	33.5493	0.9398	68	86	0.6951	0.6093	67	86	0.7252	0.6966
15	32.6696	0.9328	59	88	0.6274	0.5198	58	88	0.7370	0.6451
20	32.2144	0.9271	54	89	0.5139	0.4351	50	89	0.6045	0.6160
50	31.0675	0.8961	38	92	0.4203	0.3960	30	94	0.4826	0.4378
75	30.1511	0.8721	32	93	0.3831	0.3106	22	95	0.4163	0.3926
100	29.2537	0.8501	27	94	0.3013	0.2877	17	96	0.3937	0.3616
Frame sequence 2398, Frame size 505 KB										
5	35.3702	0.9517	104	79	0.7932	0.7213	102	80	1.0329	0.9067
10	33.7746	0.9410	81	83	0.6231	0.6134	79	84	0.8411	0.7957
15	33.0977	0.9345	69	86	0.5932	0.4929	67	87	0.7535	0.6757
20	32.6712	0.9291	64	87	0.4982	0.4036	60	89	0.6904	0.6510
50	32.2985	0.8945	42	92	0.3864	0.3289	37	93	0.5370	0.4647
75	30.2647	0.8662	32	94	0.3225	0.2854	28	94	0.4656	0.4186
100	29.2861	0.8439	29	94	0.3019	0.2978	23	95	0.4337	0.3908

Table C.2: Comparison of compression performance metrics between JPEG, JPEG2000, and the proposed methods at approximately equal compression ratios.

Method	**CR** [%]	**Size** [KB]	**PSNR** [dB]	**SSIM**	**Processing time** [s]	
					Encoding	**Decoding**
JPEG	85.7	81	26.1274	0.801	0.1921	0.3793
JPEG2000	85	72	40.09	0.9647	0.112	0.444
Proposed (single-level)	86	68	33.5493	0.9398	0.6951	0.6093
Proposed (two-level)		67			0.7252	0.6966

Table C.3: Complexity analysis metrics of JPEG encoding (processing frame sequence 1132).

Stage	**Mults.**	**Adds.**	**Bit ops.**	**Comps.**	**Shifts**	**M/R**	**M/W**	**Total ops.**
Compression Metrics								
DC shift	0	307200	0	0	0	307200	307200	921600
DCT	307200	537600	0	0	0	307200	307200	1459200
Quantization	307200	0	307200	0	307200	307200	307200	1536000
Huffman cod.	0	0	399360	337920	61440	307200	61440	1167360
Total ops.	614400	844800	706560	337920	368640	1228800	983040	5084160
Decompression Metrics								
Huffman dec.	0	0	92160	61440	61440	61440	307200	583680
Dequantization	307200	0	307200	0	307200	307200	307200	1536000
IDCT	307200	537600	0	0	0	307200	307200	1459200
Inv. DC shift	0	307200	0	0	0	307200	307200	921600
Total ops.	614400	844800	399360	61440	368640	983040	1228800	4500480

Table C.4: Complexity analysis metrics of JPEG2000 (processing frame sequence 1132).

Stage	**Mults.**	**Adds.**	**Bit ops.**	**Comps.**	**Shifts**	**M/R**	**M/W**	**Total ops.**
Compression Metrics								
DC shift	0	307200	0	0	0	307200	307200	921600
DWT	8184000	6547200	0	0	3273600	3273600	1636800	22915200
Quantization	307200	0	0	307200	307200	307200	307200	1536000
Tier1 coding	0	0	3932160	2949120	0	1310720	655360	8847360
Tier2 coding	614400	614400	3072000	0	0	614400	307200	5222400
Total ops.	9105600	7468800	7004160	3256320	3580800	5813120	3213760	39442560
Decompression Metrics								
Tier2 decoding	0	307200	1843200	0	0	307200	307200	2764800
Tier1 decoding	0	0	2621440	983040	0	655360	327680	4587520
Dequantization	307200	0	0	0	307200	307200	307200	1228800
IDWT	6547200	4910400	0	0	1636800	1636800	1636800	16368000
Inv. DC shift	0	307200	0	0	0	307200	307200	921600
Total ops.	6854400	5524800	4464640	983040	1944000	3213760	2886080	25870720

Table C.5: Complexity analysis metrics of the proposed method using a single-level encoding (processing frame sequence 1132).

Stage	**Mults.**	**Adds.**	**Bit ops.**	**Comps.**	**Shifts**	**M/R**	**M/W**	**Total ops.**
Compression Metrics								
DCT	307200	268800	307200	4800	0	307200	307200	1502400
Quantization	307200	0	307200	0	0	307264	307200	1228864
Minimization	307200	204800	102400	307200	0	307200	102400	1331200
Arithmetic coding	0	102400	819200	102400	819200	102400	102400	2048000
Total ops.	921600	576000	1536000	414400	819200	1024064	819200	6110464
Decompression Metrics								
Arithmetic decoding	0	102400	819200	102400	819200	102400	102400	2048000
Sequential search	1536000	1536000	307200	1536000	0	1536000	307200	6758400
Dequantization	307200	0	307200	4800	0	307264	307200	1233664
IDCT	307200	268800	38400	4800	0	307200	307200	1233600
Total ops.	2150400	1907200	1472000	1648000	819200	2252864	1024000	11273664

Appendix D
Chapter 5 Supplementary Results

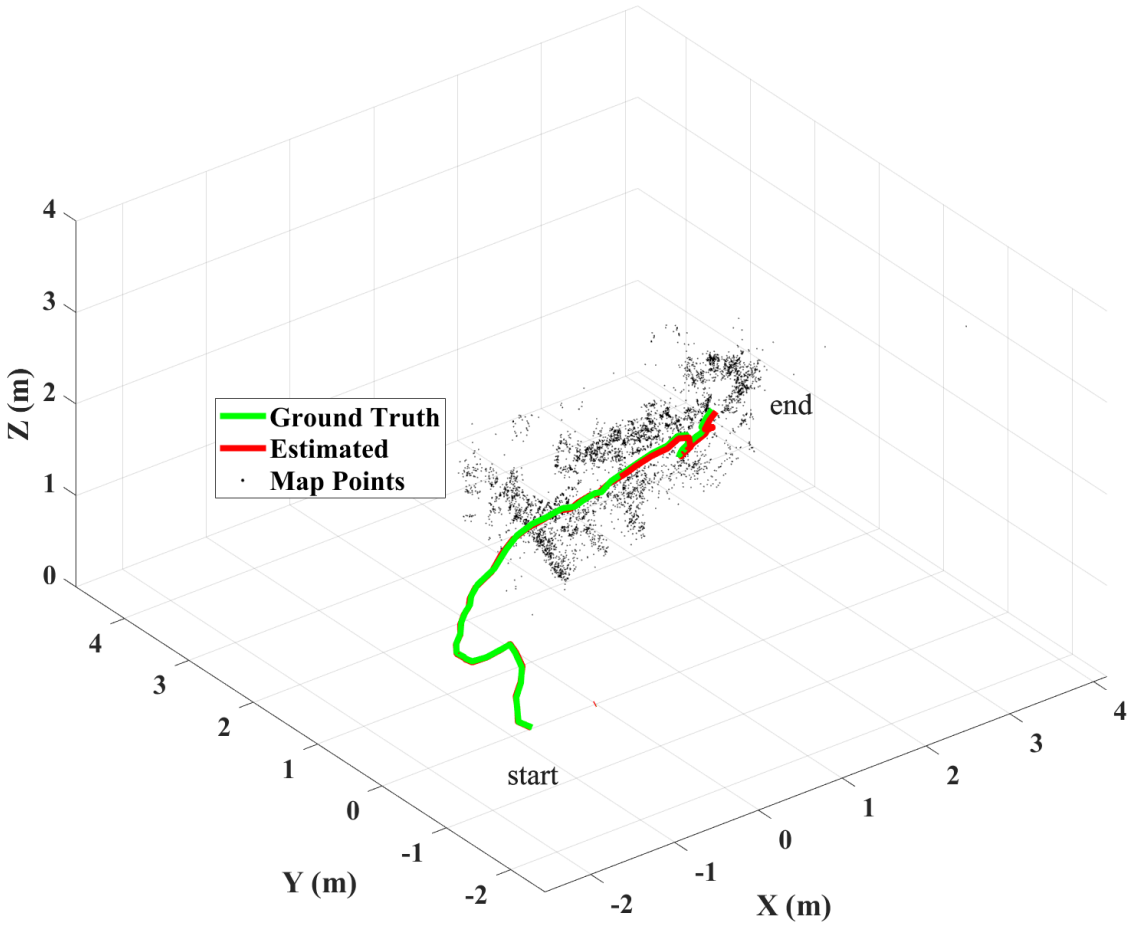


Figure D.1: Estimated trajectory of Robot-1 compared to ground truth.

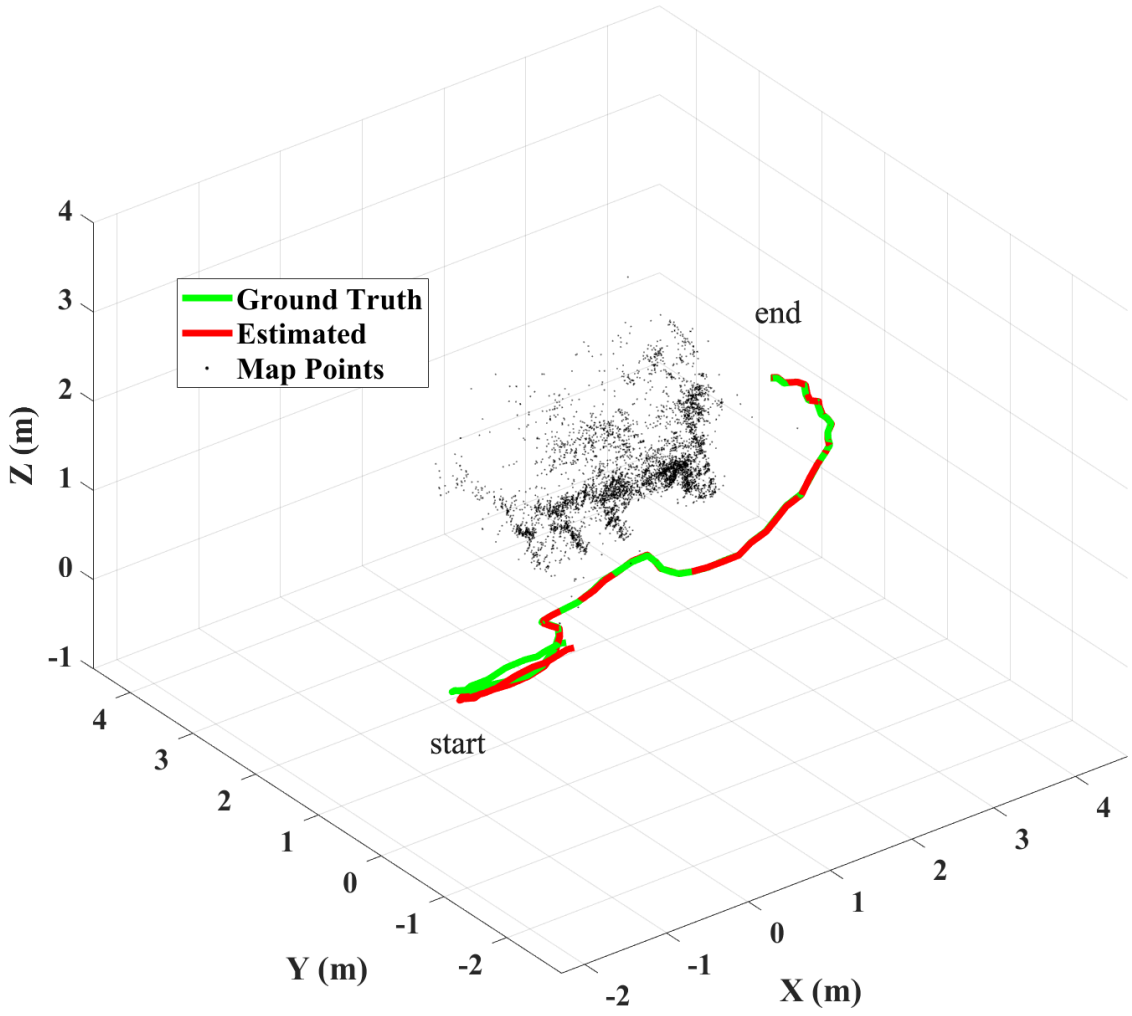


Figure D.2: Estimated trajectory of Robot-2 compared to ground truth.

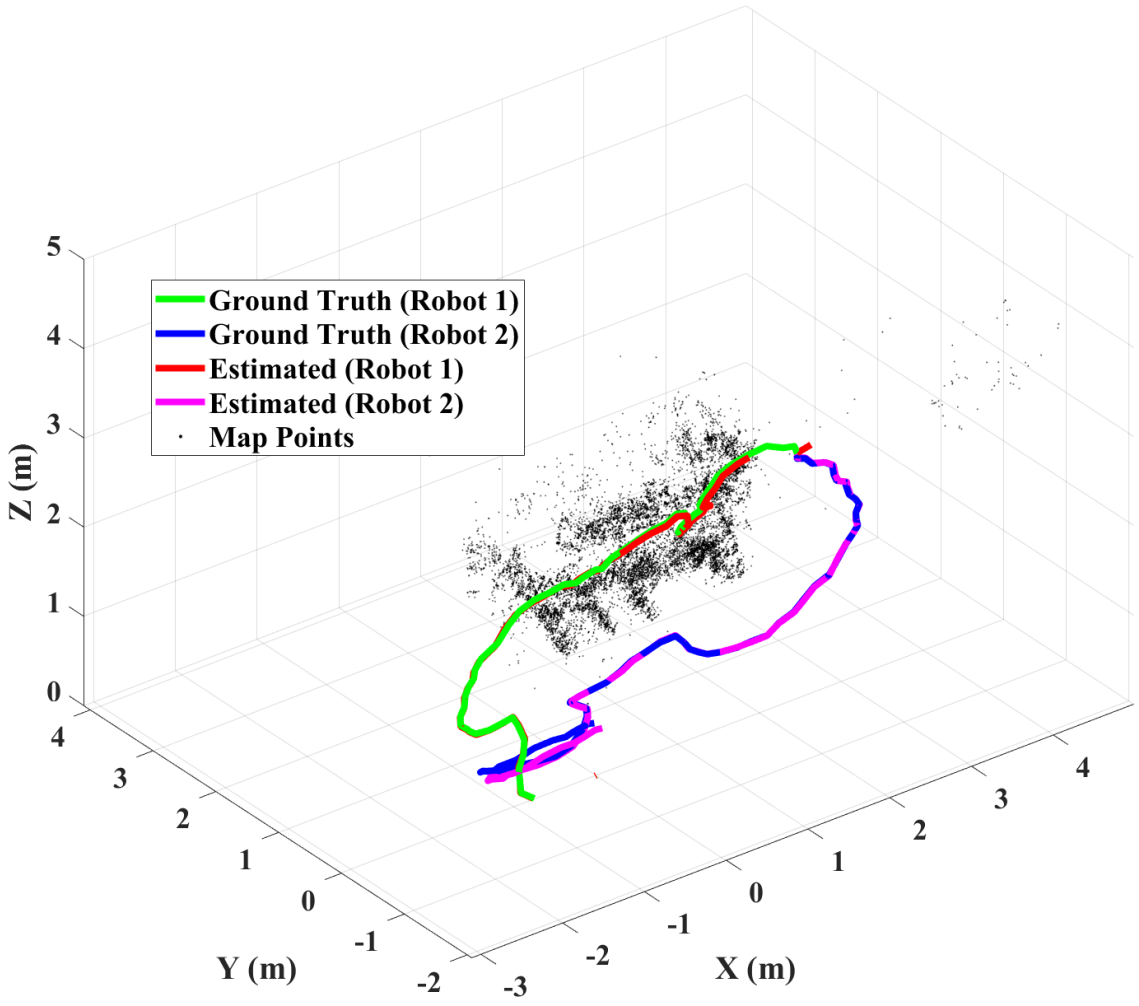


Figure D.3: Proposed multi-robot ORB-SLAM trajectory estimation using uncompressed visual data and the resulting global fused map generated from the combined robot trajectories.

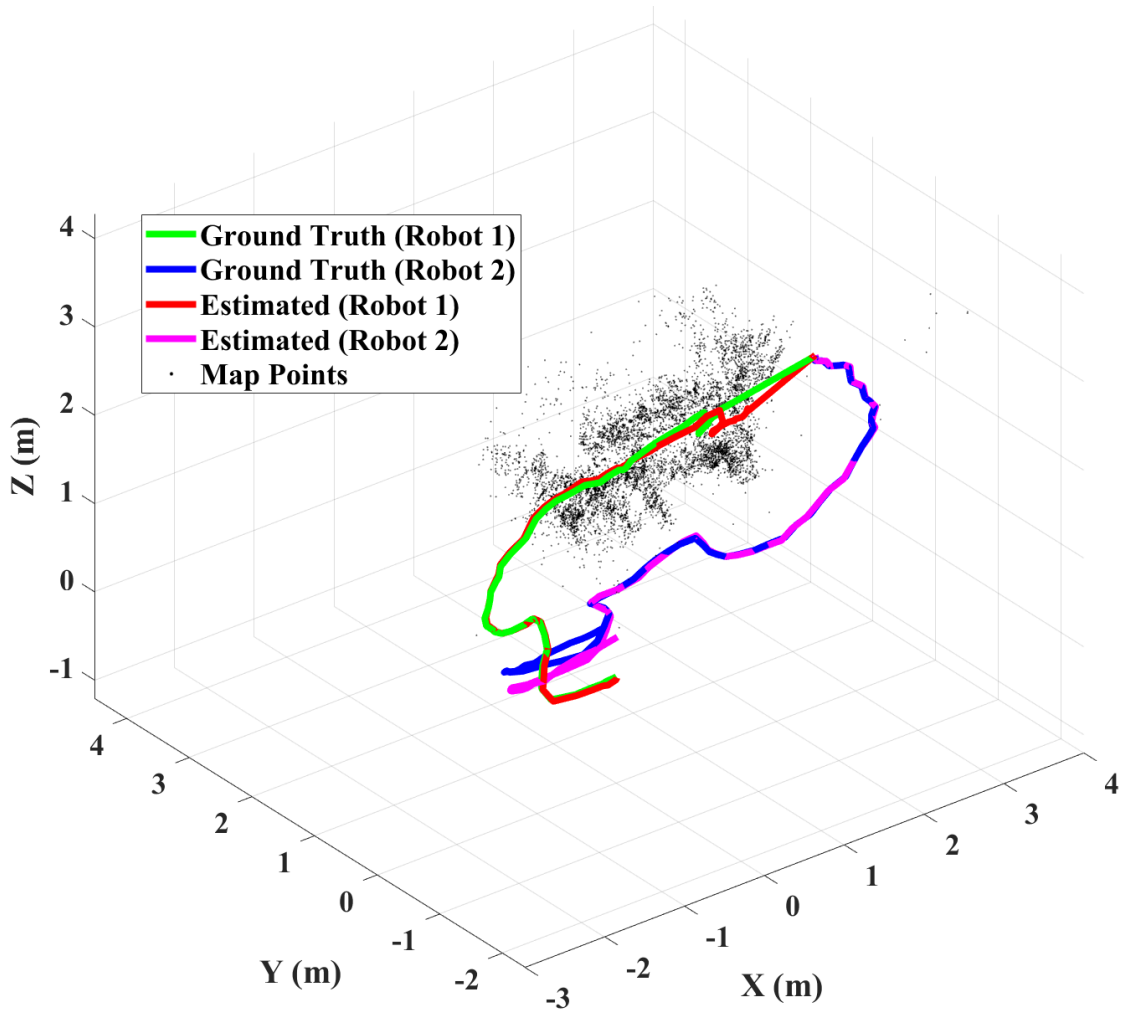


Figure D.4: Visualization of global map trajectories at high quantization (Q=100).

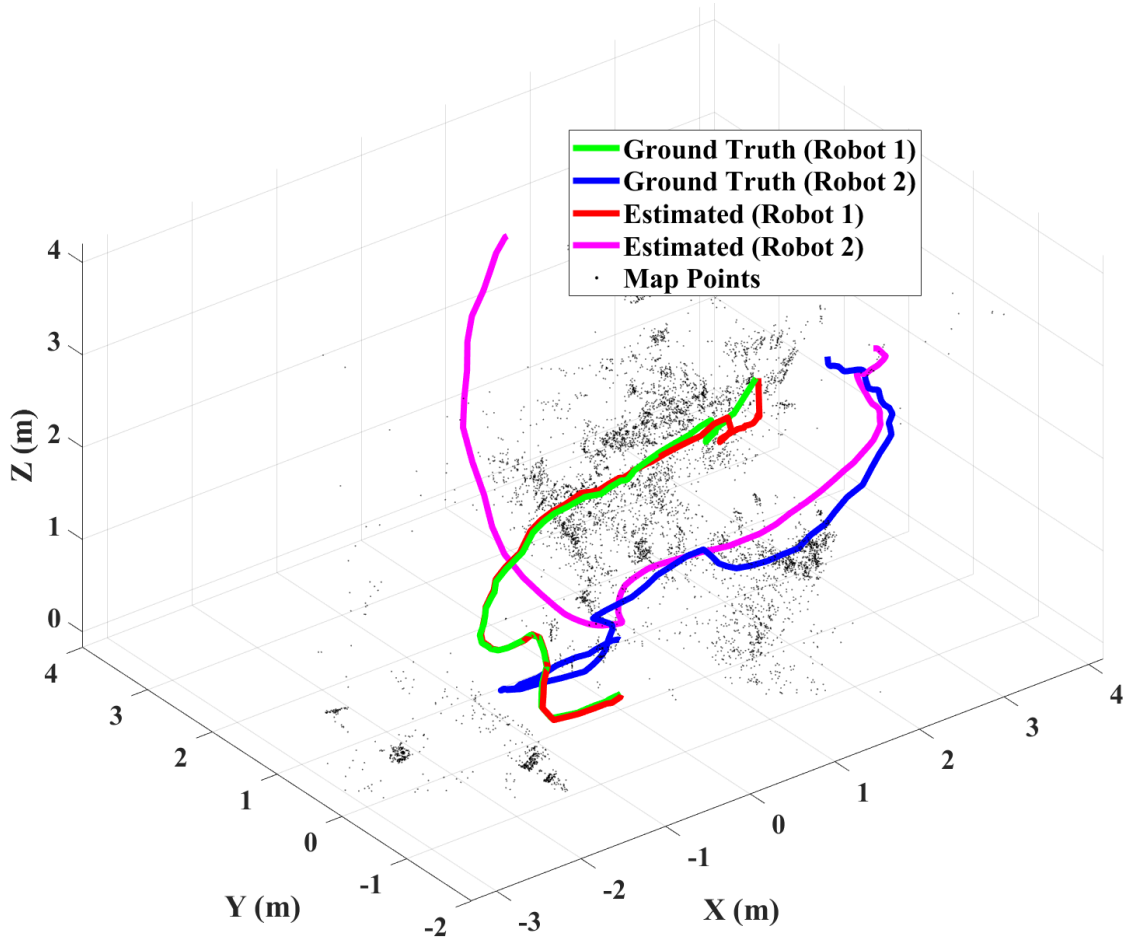


Figure D.5: Visualization of global map trajectories at severe quantization (Q=120).

Table D.1: Compression performance of the proposed methods, JPEG, and JPEG2000 for frame 2068 (900 KB).

Implementation	**Q**	**Compressed size [KB]**	**PSNR [dB]**	**SSIM**	**CR [%]**	**Enc. time [s]**	**Dec. time [s]**
			JPEG				
–	5	169	33.7052	0.9271	81.2	0.3647	0.5487
–	10	165	32.2014	0.9150	81.6	0.3165	0.5325
–	15	161	31.2274	0.8948	82.1	0.2964	0.5414
–	20	152	28.6571	0.8346	83.1	0.2571	0.5021
			JPEG2000				
–	n/a	189	41.23	0.9815	85	0.165	0.455
–	n/a	153	38.45	0.9406	88	0.124	0.432
–	n/a	45	34.71	0.9125	95	0.131	0.411
–	n/a	18	31.12	0.8849	98	0.148	0.427
			Proposed method				
Searching	25	162	30.3663	0.9032	82.0	0.4425	0.8704
Direct calculation	25	86.4	30.3663	0.9032	90.4	0.5881	0.8731
Table	25	86.4	30.3663	0.9032	90.4	0.5881	18.2558
Searching	50	135	29.4914	0.8738	85.0	0.4221	0.7321
Direct calculation	50	54.9	29.4914	0.8738	93.9	0.4843	0.6469
Table	50	54.9	29.4914	0.8738	93.9	0.4843	18.2558
Searching	75	81	28.6856	0.8451	91.0	0.3827	0.6053
Direct calculation	75	23.4	28.6856	0.8451	97.4	0.4346	0.5375
Table	75	23.4	28.6856	0.8451	97.4	0.4346	18.2558
Searching	100	21.6	27.8886	0.8157	97.6	0.3670	0.5357
Direct calculation	100	8.1	27.8886	0.8157	99.1	0.4038	0.5120
Table	100	8.1	27.8886	0.8157	99.1	0.4038	18.2558

Table D.2: Complexity analysis metrics of the proposed methods for processing a single frame (frame sequence 2068)

Stage	Multiplications	Additions	Bit operations	Comparisons	Shifts	M/R	M/W	Total Stage Operations
Compression Metrics								
DCT	307200	268800	307200	4800	0	307200	307200	1502400
Quantization	307200	0	307200	0	0	307264	307200	1228864
Minimization	307200	204800	102400	307200	0	307200	102400	1331200
Arithmetic coding	0	102400	819200	102400	819200	102400	102400	2048000
Total operations count	**921600**	**576000**	**1536000**	**414400**	**819200**	**1024064**	**819200**	**6110464**
Decompression Metrics (Searching Method)								
Arithmetic decoding	0	102400	819200	102400	819200	102400	102400	2048000
Sequential search	1536000	1536000	307200	1536000	0	1536000	307200	6758400
Dequantization	307200	0	307200	4800	0	307264	307200	1233664
IDCT	307200	268800	38400	4800	0	307200	307200	1233600
Total operations count	**2150400**	**1907200**	**1472000**	**1648000**	**819200**	**2252864**	**1024000**	**11273664**
Decompression Metrics (Direct Calculation Method)								
Arithmetic decoding	0	102400	819200	102400	819200	102400	102400	2048000
Sequential search	102400	204800	102400	0	204800	102400	307200	1024000
Dequantization	307200	0	307200	4800	0	307264	307200	1233664
IDCT	307200	268800	38400	4800	0	307200	307200	1233600
Total operations count	**716800**	**576000**	**1267200**	**112000**	**1024000**	**1536000**	**921600**	**5539264**
Decompression Metrics (Table Method)								
Arithmetic decoding	0	102400	819200	102400	819200	102400	102400	2048000
Sequential search	0	0	5242880000	5242880000	0	5243187200	307200	15729254400
Dequantization	307200	0	307200	4800	0	307264	307200	1233664
IDCT	307200	268800	38400	4800	0	307200	307200	1233600
Total operations count	**614400**	**371200**	**5244044800**	**5242992000**	**819200**	**5243904064**	**1024000**	**15733769664**

Appendix E
Chapter 6 Supplementary Material

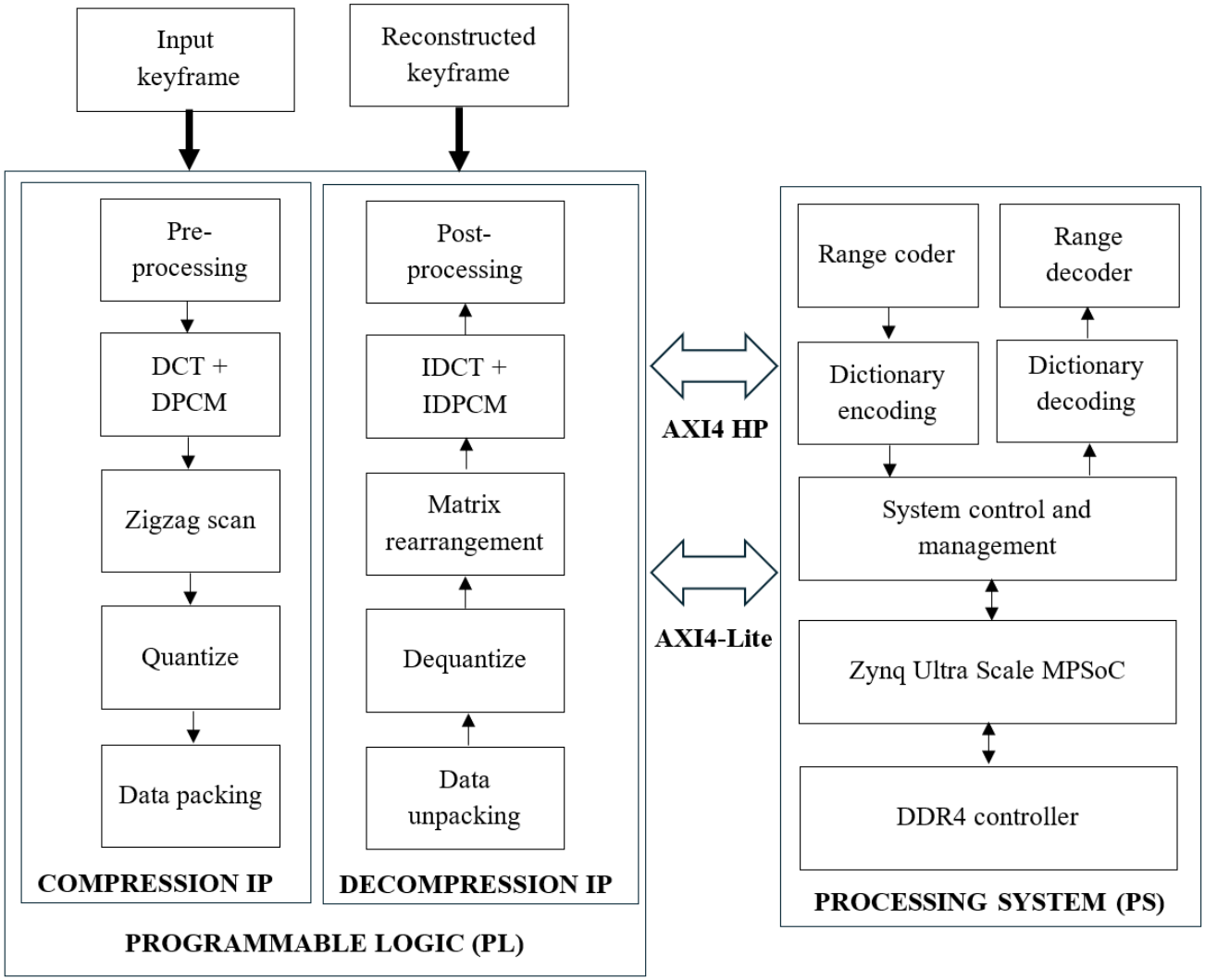


Figure E.1: Hardware–software co-design architecture of the proposed FPGA-based visual data encoding and decoding system.

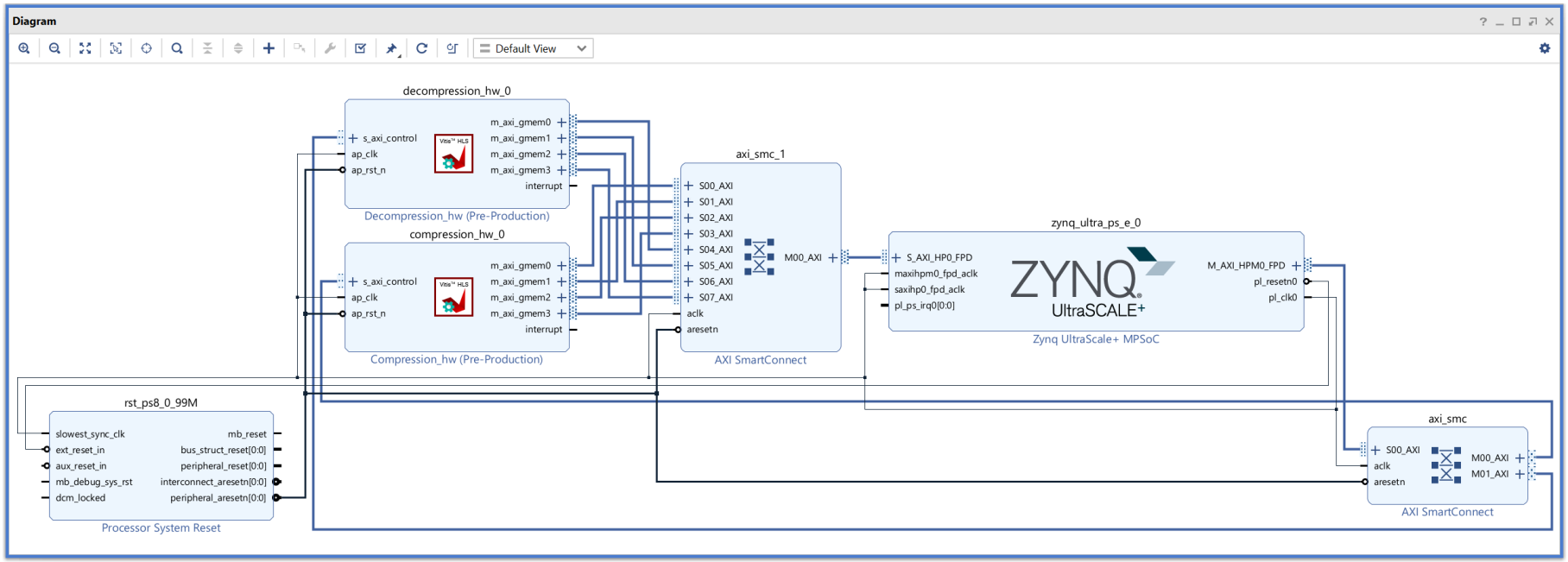


Figure E.2: Implementation of the proposed FPGA-based system using AMD Vivado.