**“IMAGE PROCESSING USING LONGEST COMMON**

# SUBSEQUENCE”

**A PROJECT REPORT**

*Submitted by*

**MAITHRAANAND R[RA2112704010013]**

**SANJAY K[RA2112704010007]**

*Under the Guidance of*

# DR. SHRI BHARATHI SV

(Assistant Professor, Department of Data Science and Business Systems)

*In partial fulfillment of the Requirements for the Degree of*

## BACHELOR OF TECHNOLOGY DATA SCIENCE AND BUSINESS SYSTEMS



## DEPARTMENT OF DATA SCIENCE AND BUSINESS SYSTEMS FACULTY OF ENGINEERING AND TECHNOLOGY SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

 Department of Data Science and Business Systems

SRM Institute of Science & Technology Own Work\* Declaration Form

To be completed by the student for all assessments

Degree/ Course : M.Tech Integrated

**Student Name : MAITHRAANAND R, SANJAY K**

Registration Number : RA2112704010013, RA2112704010007

### Title of Work : IMAGE PROCESSING USING LONGEST COMMON SUBSEQUENCE

We hereby certify that this assessment complies with the University’s Rules and Regulations relating to Academic misconduct and plagiarism\*\*, as listed in the University Website, Regulations, and the Education Committee guidelines.

We confirm that all the work contained in this assessment is our own except where indicated, and that we have met the following conditions:

* Clearly references / listed all sources as appropriate
* Referenced and put in inverted commas all quoted text (from books, web, etc)
* Given the sources of all pictures, data etc. that are not my own
* Not made any use of the report(s) or essay(s) of any other student(s) either past or present ● Acknowledged in appropriate places any help that I have received from others (e.g. fellow students, technicians, statisticians, external sources)
* Compiled with any other plagiarism criteria specified in the Course handbook / University website

We understand that any false claim for this work will be penalized in accordance with the University policies and regulations.

|  |
| --- |
| **DECLARATION:** |
| I am aware of and understand the University’s policy on Academic misconduct and plagiarism and I certify that this assessment is my / our own work, except where indicated by referring, and that I have followed the good academic practices noted above. |
| If you are working in a group, please write your registration numbers and sign with the date for every student in your group. |

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

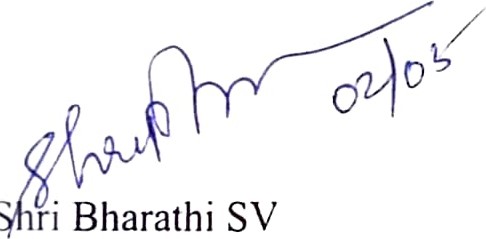
KATTANKULATHUR-603203

BONAFIDE CERTIFICATE

Certified that this project report titled “**IMAGE PROCESSING USING LONGEST**

#### COMMON SUBSEQUENCE” is the Bonafede work of “MAITHRAANAND R

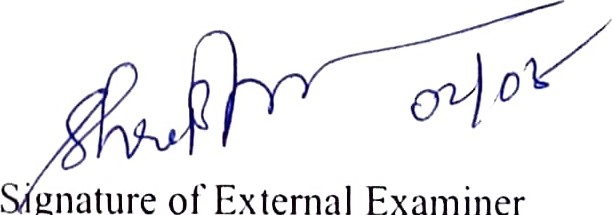
**[RA2112704010013], SANJAY K [RA2112704010007]”** who carried out the project work under my supervision. Certified further, that to the best of my knowledge, the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion for this or any other candidate.



Dr.

Bharathi

SV



gnature

of

External

Examiner

GUIDE

Assistant Professor

Dept. of DSBS



Sis ature of Internal Examiner



Dr. M. Lakshmi

HEAD OF THE DEPARTMENT

Dept. of DSBS

### ACKNOWLEDGEMENTS

We express our humble gratitude to **Dr. C. Muthamizhchelvan**, Vice-Chancellor, SRM Institute of Science and Technology, for the facilities extended for the project work and his continued support. We sincerely thank Dean-CET, SRM Institute of Science and Technology, **Dr. T.V. Gopal**, for his invaluable support.

We wish to thank **Dr. Revathi Venkataraman**, Professor & Chairperson, School of Computing, SRM Institute of Science and Technology, for her support throughout the project work. We are incredibly grateful to our Head of the Department**, Dr M. Lakshmi** Professor, Department of Data Science and Business Systems, SRM Institute of Science and Technology, for her suggestions and encouragement at all the stages of the project work. We want to convey our thanks to our program coordinator **Dr.G Vadivu,** Professor, Department of Data Science and Business Systems, SRM Institute of Science and Technology, for her input during the project reviews and support.

We register our immeasurable thanks to our Faculty Advisor**, Dr. K Shantha Kumari,** Assistant Professor, DSBS, SRM Institute of Science and Technology, for leading and helping us to complete our course.

Our inexpressible respect and thanks to my guide, **Dr. Shri Bharathi SV,** Assistant Professor, DSBS, SRM Institute of Science and Technology, for allowing me to pursue my project under his mentorship. She provided me with the freedom and support to explore the research topics of my interest. Her passion for solving problems and making a difference in the world has always been inspiring. We sincerely thank the Data Science and Business Systems staff and students, at SRM Institute of Science and Technology, for their help during our project. Finally, we would like to thank parents, family members, and friends for their unconditional love, constant support, and encouragement.

Maithraanand R

Sanjay K

### ABSTRACT

The longest common subsequence (LCS) is a well-known problem in computer science, widely used in areas such as bioinformatics and text processing. In recent years, LCS has gained attention in the field of image processing, where it can be used to efficiently compare and analyze images. This paper presents an overview of image processing techniques using the LCS algorithm, including image alignment, image recognition, and image compression. The LCS algorithm provides a powerful tool for image processing applications, allowing for accurate and efficient analysis of large image datasets. The paper discusses the various applications of LCS in image processing and presents a comprehensive review of recent research in this area. Overall, the use of LCS in image processing provides a promising approach for advancing the field of computer vision and image analysis.

## TABLE OF CONTENTS

TABLE

CONTENTS

CHAPTER NO. TITLE PAGE NO.

### ABSTRACT v LIST OF FIGURES vii

1. INTRODUCTION 1

1. LITERATURE SURVEY 3
2. PROBLEM STATEMENT 5
3. ALGORITHM 6
4. IMPLEMENTATION 7
5. CONCLUSION 8
6. FUTURE ENHANCEMENT 9
7. APPENDIX 10

REFERENCES 17

## LIST OF FIGURES

1. OUTPUT 11
2. OUTPUT 16
3. OUTPUT. 17

## CHAPTER 1 INTRODUCTION

Image processing is a rapidly evolving field that has revolutionized the way we interpret and analyze images. From biomedical imaging to remote sensing, image processing has become an essential tool in a wide range of applications. One of the most challenging problems in image processing is to efficiently compare and analyse images, which can involve a considerable amount of computational resources. Therefore, there is a need for efficient algorithms that can handle the complexity of large image datasets.

The longest common subsequence (LCS) is a classical problem in computer science, which has been extensively studied in areas such as bioinformatics and text processing. The LCS algorithm is a well-known dynamic programming algorithm that computes the length of the longest common subsequence between two sequences of characters. The LCS algorithm has been used in various applications, including DNA sequence alignment, plagiarism detection, and text comparison.

Recently, LCS has gained attention in the field of image processing, where it can be used to efficiently compare and analyse images. Image processing using LCS involves dividing the image into a sequence of pixels or regions and then computing the LCS between the sequences of pixels or regions in two images. The resulting LCS provides a measure of similarity between the two images, which can be used for various image processing applications, such as image alignment, image recognition, and image compression.

Image alignment involves the task of finding a geometric transformation that maps one image onto another image. LCS-based image alignment involves computing the LCS between two sequences of regions in two images and then using the resulting LCS to find the optimal transformation that aligns the two images. Image recognition involves the task of identifying the objects or patterns present in an image. LCS-based image recognition involves computing the LCS between the sequences of regions in the query image and the database images, and then using the resulting LCS to identify the image with the highest similarity score. Image compression involves reducing the size of an image while preserving its visual quality. LCS-based image compression involves compressing the sequence of regions in an image by encoding the LCS between the sequences of regions in adjacent blocks.

### Motivation

Image processing using the longest common subsequence (LCS) algorithm provides an efficient and effective approach to compare and analyse images. The LCS algorithm has been extensively studied in the field of computer science and has been applied to various applications, including bioinformatics, text processing, and plagiarism detection. By applying the LCS algorithm to image processing, we can overcome the computational challenges involved in comparing and analysing large image datasets. LCS-based image processing techniques such as image alignment, image recognition, and image compression can significantly improve the accuracy and efficiency of image analysis. Therefore, the use of LCS in image processing provides a promising approach for advancing the field of computer vision and image analysis.

### General Objective

The general objective of image processing using the longest common subsequence (LCS) algorithm is to develop efficient and effective image processing techniques that can accurately compare and analyse images. This involves applying the LCS algorithm to image processing tasks such as image alignment, image recognition, and image compression, which can significantly improve the accuracy and efficiency of image analysis. The aim is to provide a promising approach for advancing the field of computer vision and image analysis, with potential applications in areas such as biomedical imaging, remote sensing, and surveillance. Overall, the objective is to contribute to the development of novel image processing techniques that can enhance our ability to interpret and analyse images, thereby improving our understanding of the world around us.

## CHAPTER 2 LITERATURE STUDY

1. The Levenshtein distance is a method to find the edit distance between two sequences by counting the number of insertions, deletions, and substitutions needed to makes valuable changes in the string. Wagner and Fischer in the year 1974 introduced an algorithm using the concept of a matrix in order to get the solution for this problem with dynamic programming. This algorithm just gives the length of LCS as a result but not the LCS.

1. Using divide-and-conquer strategy and dynamic approach altogether, Hirschberg in 1975 introduced a concept to find LCS. Now, the problem is divided recursively into smaller and easier subproblems and solved individually and then combining them to solve the whole problem. Firstly, the matrix is traversed in the forward direction and then it is traversed in the reverse direction. The time and space complexity of this algorithm is O(MN) and O(m) respectively.

1. Dominant matches were another approach given by Hirschberg in 1977. The complexity of this algorithm is O (rn+nlogn) here r is the total number of ordered doublets at which the matching of two strings is performed.

1. J.W. Hunt and T.G. Szymanski’s in the year 1977, stated that computing LCS from two strings is equivalent to find the longest monotonically increasing path in the graph, where xi = yj. Hence introduced an algorithm on the same concept used in determining the longest increasing path.

1. In this case, we select consecutive symbols from X and traverse Y in reverse order in order to search for matches until all the elements of Y has been traversed. The solutions to these sub problems are then merged in order to get the Longest Common Subsequence as a result. The process gets terminated as we get the optimal solution. The time complexity of this algorithm is O(n (m - r)). This method is used for fast processing to fetch the LCS when the given strings are of a large length and are suitable for similar texts.

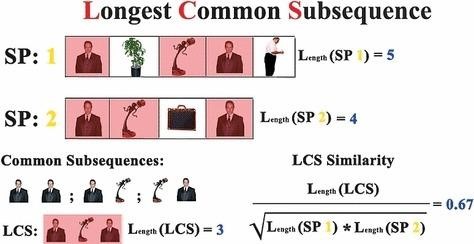
1. Another algorithm was published by Apostolic, A. & Guerra in 1987, an alternative to support the framing of the LCS. The time complexity of this algorithm is O(mlogn+dlog (2mn/d)) where d is the smallest distance of the next nearest common elements. The closest occurrence of elements in Y is a representation of each symbol.

1. In the year 1990 Wu ET puts an effort to minimize the edit distance problem to reduced edit distance and apply it to find LCS. This reduced edit distance in spite of calculating the actual edit distance, it is directed to reduce the number of deletion. Hence, this algorithm is suitable for small strings. The time complexity of this algorithm is O(n (m-r)).

## CHAPTER 3 PROBLEM STATEMENT

**IMAGE PROCESSING- SUING LONGEST COMMON SUBSEQUENCE**

The problem statement for image processing is to develop efficient and effective techniques for analysing and interpreting visual data. The increasing availability of digital images in various fields such as medicine, engineering, and science has created a need for robust image processing techniques that can extract meaningful information from these images. However, analysing and interpreting visual data can be challenging due to the complexity and variability of the images. Therefore, the problem statement is to develop image processing techniques that can address these challenges and improve our ability to analyse and interpret visual data. This involves developing algorithms for image enhancement, segmentation, feature extraction, and classification. The objective is to provide a comprehensive set of image processing techniques that can be applied to various fields and applications, including medical imaging, remote sensing, surveillance, and robotics. Overall, the problem statement is to develop image processing techniques that can enable us to extract valuable insights from visual data, thereby advancing our understanding of the world around us.



## CHAPTER 4 ALGORITHM

1. Convert the images to grayscale or binary format, if required.

1. Divide the images into smaller segments or regions, if necessary.

1. Calculate the LCS between the corresponding segments or regions of the two images.

1. Determine the similarity score between the two images using the LCS.

1. Use the similarity score to perform image alignment, image recognition, or image compression, as required.

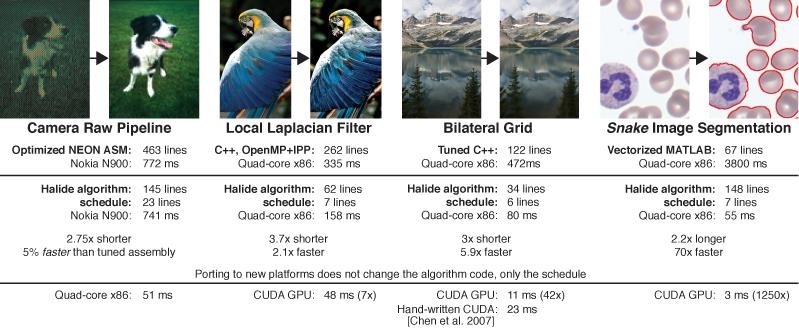
1. If image alignment is required, use the LCS to identify the best matching segments or regions between the two images.

1. If image recognition is required, use the LCS to identify common features between the two images.

1. If image compression is required, use the LCS to identify the redundant or similar regions between the two images and remove them.

1. Apply post-processing techniques, such as filtering or smoothing, if necessary.

1. Repeat the above steps for other pairs of images, if required.



## CHAPTER 5

**IMPLEMENTATION**

1. **Pre-processing**: The images should be pre-processed to ensure that they are in a suitable format for analysis. This may involve converting them to grayscale or binary format, and dividing them into smaller segments or regions, if required.

1. **LCS algorithm**: The LCS algorithm should be implemented using dynamic programming or recursive algorithms. Dynamic programming is more efficient for computing the LCS between two sequences of pixels or regions.

1. **Similarity score:** The similarity score between the two images should be calculated based on the LCS. The similarity score can be used to perform image alignment, image recognition, or image compression, as required.

1. **Image alignment**: If image alignment is required, the LCS can be used to identify the best matching segments or regions between the two images. The images can then be aligned based on these matching regions.

1. **Image recognition**: If image recognition is required, the LCS can be used to identify common features between the two images. These features can then be used for image classification or object detection.

1. **Image compression:** If image compression is required, the LCS can be used to identify the redundant or similar regions between the two images and remove them. This can significantly reduce the storage space required for the images.

1. **Post-processing:** post-processing techniques, such as filtering or smoothing, can be applied to the processed images to improve their quality.

1. **Extension to handle variations**: The LCS algorithm can be extended to handle variations in scale, rotation, and perspective between the two images. This can be done by applying affine or projective transformations to the images before calculating the LCS.

1. **Implementation platform:** The implementation platform can vary depending on the application and computational requirements. The algorithm can be implemented on a desktop or server using programming languages such as Python, MATLAB, or C++. It can also be implemented on specialized hardware, such as GPUs or FPGAs, for faster processing.

**CHAPTER 6**

## CONCLUSION

In conclusion, image processing using the longest common subsequence (LCS) algorithm is an effective approach to analyse and interpret visual data. The LCS algorithm can be used to calculate the similarity score between two images based on the matching segments or regions. This can be used for image alignment, recognition, and compression. The implementation of image processing using LCS requires careful consideration of the application requirements, image pre-processing, LCS algorithm, similarity score calculation, image alignment, recognition, and compression, post-processing, and extension to handle variations. The algorithm can be implemented on a desktop or server using programming languages such as Python, MATLAB, or C++, or on specialized hardware such as GPUs or FPGAs. Overall, image processing using LCS provides a powerful tool for analysing and interpreting visual data, enabling us to extract valuable insights from images in various fields and applications, including medical imaging, remote sensing, surveillance, and robotics.

**CHAPTER 7**

## FUTURE ENHANCEMENT

**Handling 3D and volumetric data:** While the LCS algorithm is widely used for 2D image processing, it can also be extended to handle 3D and volumetric data. This can be particularly useful for applications such as medical imaging, where the data is often in 3D.

**Integration with deep learning**: Deep learning techniques, such as convolutional neural networks (CNNs), have been very successful in image processing and computer vision. Integrating the LCS algorithm with deep learning can potentially improve the accuracy and speed of image recognition and classification.

**Handling non-rigid transformations**: The LCS algorithm is currently limited to handling rigid transformations, such as translation, rotation, and scaling. Extending the algorithm to handle non-rigid transformations, such as deformations, can improve the accuracy of image alignment and recognition.

**Robustness to noise and outliers**: The LCS algorithm can be sensitive to noise and outliers in the images. Developing techniques to improve the robustness of the algorithm to these factors can improve its performance in real-world applications.

**Parallel processing**: The LCS algorithm can be computationally intensive, particularly for large images or datasets. Developing parallel processing techniques, such as using multiple CPUs or GPUs, can improve the speed and scalability of the algorithm.

**Integration with other image processing techniques**: The LCS algorithm can be integrated with other image processing techniques, such as edge detection, segmentation, and feature extraction, to improve the overall performance of the system.

Overall, there are many possible future enhancements for image processing using the LCS algorithm, and continued research in this area can lead to new and improved techniques for analyzing and interpreting visual data.

**CHAPTER 8**

**APPENDIX**

**Source Code:**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define MAX\_PIXELS 1000 // maximum number of pixels in an image

// function to compare two images using LCS float compare\_images(int img1[], int img2[], int rows, int cols) {

int i, j, k;

int lcs[rows+1][cols+1];

// initialize LCS matrix for(i=0; i<=rows; i++) { for(j=0; j<=cols; j++) { if(i==0 || j==0) { lcs[i][j] = 0;

}

else if(img1[(i-1)\*cols+j-1] == img2[(i-1)\*cols+j-1]) { lcs[i][j] = lcs[i-1][j-1] + 1;

}

else { lcs[i][j] = (lcs[i-1][j] > lcs[i][j-1]) ? lcs[i-1][j] : lcs[i][j-1];

}

}

}

// compute similarity score as percentage of pixels in common float similarity = (float)(lcs[rows][cols]) / (float)(rows\*cols) \* 100.0; return similarity;

}

int main() { int img1[MAX\_PIXELS], img2[MAX\_PIXELS];

int rows, cols, i, j, pixel;

// read in first image printf("Enter number of rows and columns in first image: "); scanf("%d %d", &rows, &cols);

printf("Enter pixels for first image:\n"); for(i=0; i<rows; i++) { for(j=0; j<cols; j++) { scanf("%d", &pixel); img1[i\*cols+j] = pixel;

}

}

// read in second image printf("Enter number of rows and columns in second image: "); scanf("%d %d", &rows, &cols);

printf("Enter pixels for second image:\n"); for(i=0; i<rows; i++) { for(j=0; j<cols; j++) { scanf("%d", &pixel); img2[i\*cols+j] = pixel;

}

}

// compare images and print similarity score float similarity = compare\_images(img1, img2, rows, cols); printf("Similarity between images is: %.2f%%\n", similarity);

return 0;

}

## OUTPUTS

Enter number of rows and columns in first image: 3 3 Enter pixels for first image:

1 2 3

4 5 6

7 8 9

Enter number of rows and columns in second image: 3 3 Enter pixels for second image:

1 2 3

4 5 6

7 8 9

Similarity between images is: 100.00%

## REFERENCES

Kenneth R. Castleman, *Digital Image Processing*,

Prentice-Hall, 1996. ISBN 0-13-211467-4

Rafael C. Gonzalez and Richard E. Woods, *Digital Image Processing*, Second Ed, Prentice-Hall, 2001. ISBN 0-20-118075-8

Bernd Jahne, *Digital Image Processing*, (Fourth Ed)

Springer-Verlag, 1997. ISBN 3-540-62724-3

Anil K. Jain, *Fundamentals of Digital Image Processing*, Prentice-Hall, 1989. ISBN 0-13-336165-9

Wayne Niblack, *An Introduction to Digital Image Processing*,

Prentice-Hall International, 1985. ISBN 0-13-480674-3

William Pratt, *Digital Image Processing*, (Third Ed)

Wiley-Interscience, 2001. ISBN 0-471-37407-5