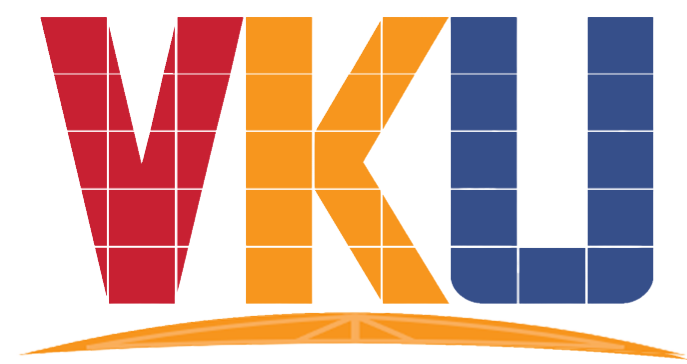


**ĐẠI HỌC ĐÀ NẴNG**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**VÀ TRUYỀN THÔNG VIỆT - HÀN**



XỬ LÝ TÍN HIỆU SỐ

**Tìm hiểu kỹ thuật nén ảnh sử dụng phương pháp SVD**

**Giảng viên hướng dẫn :** TS. Phan Thị Lan Anh

**Sinh viên thực hiện :** Bùi Văn Vạn Quý

Nguyễn Bình Minh

Trương Thành Quý

Nguyễn Phan Thành Công

**Lớp :** 18IT5

**Khoá :** 2018 - 2023

*Đà Nẵng, tháng 6 năm 2021*

**MỤC LỤC**

[LỜI CẢM ƠN 4](#_Toc73568377)

[LỜI MỞ ĐẦU 5](#_Toc73568378)

[CHƯƠNG 1: GIỚI THIỆU CHUNG 6](#_Toc73568379)

[1.1 Xử lý tín hiệu số 6](#_Toc73568380)

[1.1.1 Khái niệm 6](#_Toc73568381)

[1.1.2 Ưu điểm 6](#_Toc73568382)

[1.1.3 Ứng dụng 7](#_Toc73568383)

[1.2 Tín hiệu 8](#_Toc73568384)

[1.2.1 Tín hiệu liên tục 8](#_Toc73568385)

[1.2.2 Tín hiệu rời rạc 8](#_Toc73568386)

[1.2.3 Tín hiệu tương tự 9](#_Toc73568387)

[1.2.4 Tín hiệu số 9](#_Toc73568388)

[1.3 Phân tích tín hiệu trong miền tần số 9](#_Toc73568389)

[CHƯƠNG 2: Nén ảnh bằng phương pháp SVD 10](#_Toc73568390)

[2.1 Nén ảnh 10](#_Toc73568391)

[2.2 Khái niệm chuẩn nén ảnh số bằng SVD 10](#_Toc73568392)

[2.3 Giới thiệu về sử dụng python: 12](#_Toc73568393)

[CHƯƠNG 3: KẾT QUẢ MÔ PHỎNG BÀI TẬP 14](#_Toc73568394)

[3.1 Phần mềm mô phỏng: 14](#_Toc73568395)

[3.2 Kết quả 15](#_Toc73568396)

[3.3 Source Code : 16](#_Toc73568397)

**NHẬN XÉT CỦA GIẢNG VIÊN**

................................................................................................................................ ................................................................................................................................ ................................................................................................................................ ................................................................................................................................ ................................................................................................................................ ................................................................................................................................ ................................................................................................................................ ................................................................................................................................ ................................................................................................................................ ................................................................................................................................ ................................................................................................................................ ................................................................................................................................ ................................................................................................................................ ................................................................................................................................ ................................................................................................................................ ................................................................................................................................ ................................................................................................................................ ................................................................................................................................ ................................................................................................................................ ................................................................................................................................ ................................................................................................................................

**Ngày 4 tháng 6 năm 2021**

**Giảng viên**

# LỜI CẢM ƠN

Để hoàn thành tốt được đề tài bài tập lớn lần thì này chúng em xin cảm ơn tới quý thầy cô Trường Đại học Công nghệ thông tin và truyền thông Việt – Hàn đã tạo điều kiện cho nhóm chúng em nghiên cứu và hoàn thành đề tài. Tiếp đến, chúng em xin chân trọng cảm ơn giảng viên hướng dẫn nhóm chúng em cô Phan Thị Lan Anh đã tận tình giúp đỡ, chỉ bảo hướng dẫn chúng em trong suốt quá trình thực hiện đồ án (Bài tập lớn) môn học.

Vì thời gian và năng lực còn có hạn chế nên không thể tránh khỏi những sai sót trong khi thực hiện đề tài nghiên cứu của mình. Rất mong được sự góp ý bổ sung của thầy/cô để đề tài của chúng em ngày càng hoàn thiện hơn.

Xin chân thành cảm ơn !

# LỜI MỞ ĐẦU

Trong môn học Xử lý tín hiệu số, nhờ làm bài tập và học lý thuyết về môn này nên chúng em nắm được những kiến thức cơ bản về xử lý tín hiệu số, các phép toán về xử lý tín hiệu, trên cơ sở đó chúng em có thể tự mình sử dụng được ngôn ngữ python và viết các chương trình bằng ngôn ngữ python. Mặc dù đã cố gắng rất nhiều song do hiểu biết và kiến thức còn hạn chế, kinh nghiệm còn ít và trong quá trình làm đồ án môn học và sử dụng chương trình chúng em không thể tránh khỏi những sai xót. Mong nhận được sự chỉ dẫn của thầy cô giáo và đóng góp ý kiến của các bạn để em rút kinh nghiệm trong những sau cũng như những môn khác.

Đồ án môn học Xử lý tín hiệu số lần này hoàn thành không những giúp chúng em có thêm kiến thức hơn về môn học này mà còn giúp chúng em có một phương pháp làm việc mới, chủ động hơn, linh hoạt hơn và đặc biệt hơn là phương pháp làm việc mới. Quá trình làm bài báo cáo này là một khoảng thời gian thực sự có ích cho cả nhóm chúng em về nhiều mặt.

Cuối cùng, nhóm em xin chân thành cảm ơn đến cô Phan Thị Lan Anh đã tạo mọi điều kiện, giúp đỡ nhóm có đủ khả năng để hoàn thành đồ án môn học Xử lý tín hiệu số.

# CHƯƠNG 1: GIỚI THIỆU CHUNG

## 1.1 Xử lý tín hiệu số

### 1.1.1 Khái niệm

Xử lý tín hiệu số (Digital Signal Processing - DSP) là việc xử lý những tín hiệu đã được biểu diễn dưới dạng chuỗi những dãy số. Xử lý tín hiệu số và [xử lý tín hiệu tương tự](https://vi.wikipedia.org/w/index.php?title=X%E1%BB%AD_l%C3%BD_t%C3%ADn_hi%E1%BB%87u_t%C6%B0%C6%A1ng_t%E1%BB%B1&action=edit&redlink=1) là 2 phần của [xử lý tín hiệu](https://vi.wikipedia.org/wiki/X%E1%BB%AD_l%C3%BD_t%C3%ADn_hi%E1%BB%87u).

Công nghệ xử lý tín hiệu số là công nghệ bùng nổ nhanh chóng trong ngành công nghiệp điện tử và viễn thông hiện nay. Xử lý tín hiệu số có nhiều ứng dụng đa dạng, ví dụ như trong lĩnh vực điện tử y sinh, trong điều chỉnh động cơ diesel, xử lý thoại, các cuộc gọi điện thoại khoảng cách xa, [xử lý tiếng nói](https://vi.wikipedia.org/wiki/X%E1%BB%AD_l%C3%BD_ti%E1%BA%BFng_n%C3%B3i), [xử lý âm thanh](https://vi.wikipedia.org/w/index.php?title=X%E1%BB%AD_l%C3%BD_%C3%A2m_thanh&action=edit&redlink=1), và tăng cường chất lượng hình ảnh và truyền hình. Các công nghệ nén [MPEG](https://vi.wikipedia.org/wiki/MPEG) hay [WMV](https://vi.wikipedia.org/wiki/Windows_Media_Player) hiện nay đều dựa trên tiến bộ của công nghệ xử lý tín hiệu số.

### 1.1.2 Ưu điểm

Xử lý tín hiệu số là môn học đề cập đến phép xử lý các dãy số để có được các thông tin cần thiết như phân tích, tổng hợp mã hóa, biến đổi tín hiệu sang dạng mới phù hợp với hệ thống. So với xử lý tín hiệu tương tự, xử lý tín hiệu số có nhiều ưu điểm như :

* Độ chính xác cao, sao chép trung thực, tin cậy.
* Tính bền vững : Không chịu nhiều ảnh hưởng của nhiệt độ hay thời gian.
* Linh hoạt và mềm dẻo : Thay đổi phần mềm có thể thay đổi tính năng của phần cứng.
* Thời gian thiết kế nhanh, ngày càng hoàn thiện và có độ tích hợp cao

### 1.1.3 Ứng dụng

DSP (công nghệ xử lý tín hiệu số) là một công nghệ được sử dụng để thiết lập các vị trí lọc khác nhau và nhằm tránh can nhiễu. Các bộ lọc âm thanh audio tiêu chuẩn đưa ra một dải audio nhất định được gọi là dải thông. Để tránh can nhiễu giữa các kênh gần kề, máy thu tín hiệu analog truyền thống kết hợp với các bộ lọc dải hẹp cho phép nghe được tín hiệu ở dải thông hẹp hơn. Với dải thông hẹp, audio ở dải hẹp hơn từ các tín hiệu khác có thể ảnh hưởng đến tín hiệu mà bạn đang nghe. Chỉ có một vấn đề là do các vị trí bộ lọc hẹp nên các audio ở dải hẹp hơn có thể đi qua và tín hiệu sẽ phát ra tiếng như bị tắc. Một số bộ lọc CW dải cực hẹp đi qua audio quá nhỏ đến nỗi gần như không có tác dụng đối với các truyền dẫn thoại.

Hiện nay có một số xu hướng trong điều khiển động cơ. Trong một số trường hợp, vấn đề cần quan tâm chỉ đơn giản là điều khiển tốc độ, gia tốc, mômen hoặc các thuộc tính khác của động cơ trên cơ sở tín hiệu điều khiển đầu vào là từ phía con người, chẳng hạn như qua bảng điều khiển.

Texas Instruments (TI) cũng bắt đầu tham dự thị trường động cơ công nghiệp. TI thống lĩnh thị trường bộ điều khiển truyền động cho đĩa cứng nhưng đã giới thiệu chip mới nhắm vào điều khiển động cơ công nghiệp vào cuối năm sau đó. Bộ điều khiển DSP TMS320C240 tổ hợp lõi C2xLP của công ty với bộ quản lý sự kiện tối ưu hóa điều khiển chuyển động, các bộ biến đổi A/D kép và các ngoại vi kiểu MCU. Sản phẩm với bộ nhớ flash đã hiện hữu cho các mục đích phát triển. Hiện nay Hãng ND-Tech là nhà sản xuất, cung cấp và phát triển sản phẩm bộ xử lý tín hiệu số (phần cứng và phần mềm) trên nền hệ thống DSP TMS320C6000 của hãng Texas Instruments Incorporated – Mỹ.

Công nghiệp giải trí: âm nhạc (số) Mp3, Mp4, nhạc trực tuyến, ...

Xử lý ảnh: Nhận dạng ảnh, cải thiện chất lượng ảnh, nén dữ liệu ảnh

Xử lý tiếng nói: Nhận dạng và tổng hợp tiếng nói, mã hoá tiếng nói, ...

Truyền thông: Nén số liệu

## Tín hiệu

Tín hiệu là một dạng vật chất có một đại lượng vật lý được biến đổi theo quy luật của tin tức

* Tín hiệu âm thanh, song âm, tín hiệu điện (thời gian)
* Tín hiệu hình ảnh (không gian)

Về mặt toán học tín hiệu được coi là hàm của một hay nhiều biến độc lập.

***Ví dụ:*** Tín hiệu âm thanh là sự biến thiên của áp suất theo thời gian P(t) hoặc cũng có thể coi tín hiệu âm thanh là sự biến thiên áp suất theo không gian P(x,y,z).

### 1.2.1 Tín hiệu liên tục

Tín hiệu liên tục (về mặt thời gian) là tín hiệu mang giá trị thực (hoặc phức) xác định với mọi thời điểm trong một khoảng thời gian, trường hợp phổ biến nhất là một khoảng thời gian vô hạn.

Tín hiệu theo thời gian liên tục (tín hiệu liên tục, còn gọi là tín hiệu tương tự)

* Có thể thay đổi tại bất kỳ thời điểm nào.
* Thường có bản chất tự nhiên (ví dụ: nhiệt độ).

### 1.2.2 Tín hiệu rời rạc

Tín hiệu rời rạc (về mặt thời gian) là tín hiệu chỉ xác định trên một tập rời rạc của thời gian (một tập những thời điểm rời rạc). Dưới dạng toán học, tín hiệu rời rạc mang giá trị thực (hoặc phức) có thể được xem là một hàm liên kết tương ứng từ tập số tự nhiên đến tập số thực (hoặc phức).

Tín hiệu theo thời gian rời rạc (tín hiệu rời rạc)

* Chỉ thay đổi tại những thời điểm nhất định.
* Thường liên quan tới các hệ thống nhân tạo.

### 1.2.3 Tín hiệu tương tự

Tín hiệu tương tự là tín hiệu có giá trị thay đổi liên tục theo thời gian

### 1.2.4 Tín hiệu số

Tín hiệu số là tín hiệu đã được [lấy mẫu](https://vi.wikipedia.org/w/index.php?title=L%E1%BA%A5y_m%E1%BA%ABu&action=edit&redlink=1) và [lượng tử hóa](https://vi.wikipedia.org/wiki/L%C6%B0%E1%BB%A3ng_t%E1%BB%AD_h%C3%B3a).

## Phân tích tín hiệu trong miền tần số

Tín hiệu thường được phân tích trong miền tần số. Phương pháp này áp dụng cho các loại tín hiệu, cả tín hiệu liên tục hay rời rạc theo thời gian. Nghĩa là khi cho một tín hiệu đi qua một hệ thống tuyến tính, không đổi theo thời gian, thì [phổ tần số](https://vi.wikipedia.org/w/index.php?title=Ph%E1%BB%95_t%E1%BA%A7n_s%E1%BB%91&action=edit&redlink=1) của tín hiệu đầu ra sẽ bằng tích của phổ tần số của tín hiệu đầu vào và [đáp ứng xung](https://vi.wikipedia.org/wiki/%C4%90%C3%A1p_%E1%BB%A9ng_xung) của [hệ thống](https://vi.wikipedia.org/wiki/H%E1%BB%87_th%E1%BB%91ng). Một số phương pháp chuyển đổi tín hiệu giữa miền thời gian và miền tần số là:

* [Fourier](https://vi.wikipedia.org/wiki/Joseph_Fourier)
* [Laplace](https://vi.wikipedia.org/wiki/Pierre-Simon_Laplace)
* [Hilbert](https://vi.wikipedia.org/w/index.php?title=Hilbert&action=edit&redlink=1)
* [Biến đổi Z](https://vi.wikipedia.org/wiki/Bi%E1%BA%BFn_%C4%91%E1%BB%95i_Z)
* [Wavelet](https://vi.wikipedia.org/w/index.php?title=Wavelet&action=edit&redlink=1)

# CHƯƠNG 2: Nén ảnh bằng phương pháp SVD

## 2.1 Nén ảnh

* Nén ảnh là một công nghệ nhằm mã hóa các loại ảnh kỹ thuật số để làm giảm số bít biểu diễn ảnh giúp làm giảm bộ nhớ và giá trị khi truyền ảnh.
* Nén dữ liệu ảnh là biến đổi dòng thông tin từ ảnh thành đoạn mã, nhằm giảm độ dư thừa thông tin (không gian và thời gian, độ dư thừa phổ và độ dư thừa do độ cảm thụ).
* Dung lượng thông tin bao gồm: lưu trữ và truyền (tốc độ dòng bít).
* Lượng thông tin thu được sau khi nén thường nhỏ hơn dữ liệu gốc.
* Tính hiệu quả dựa trên: chi phí về thời gian, không gian và độ phức tạp khi tính toán.

## 2.2 Khái niệm chuẩn nén ảnh số bằng SVD

Ảnh được sử dụng nhiều trong các ứng dụng của máy tính nhưng lại khó khăn trong việc lưu trữ hoặc truyền nhận do kích thước thường khá lớn. Người ta đề xuất ra nhiều kĩ thuật nén ảnh để loại bỏ thông tin ít ảnh hưởng đến ảnh nhưng lại chiếm rất nhiều dung lượng lưu trữ. Nén ảnh có thể đạt được bằng cách sử dụng kỹ thuật SVD với ưu điểm là khả năng tương thích với sự thay đổi thống kê của ảnh và khả năng thực hiện trên ma trận có kích thước tùy ý. Cách thức SVD nén ảnh như sau:

Cho A là ảnh xám, hoặc một trong 3 thành phần ảnh màu, được biểu diễn bởi ma trận *m×n*. Sử dụng biến đổi SVD của A chúng ta có thể biểu diễn ma trận ảnh



với r là bậc của ma trận *A* và Σ là ma trận đường chéo m×n gồm các trị đơn  của *A*, . Trong nén ảnh, ta không cần lưu trữ hay truyền toàn bộ dữ liệu ảnh nhưng vẫn lưu trữ hoặc truyền đủ những thông tin quan trọng giúp khôi phục được ảnh. Biến đổi SVD thích hợp với điều này vì ta có thể xấp xỉ *A* bằng cách bỏ đi các thành phần ảnh tương ứng với các trị đơn  có giá trị nhỏ, do đó ảnh sẽ không mất nhiều thông tin khi thay chúng bằng 0. Từ đó, ảnh xấp xỉ được tạo ra như sau:

 .

Điều đó có nghĩa là thay vì toàn bộ *A*, chúng ta có thể chỉ cần lưu trữ và truyền nhận  và . Hình 1 minh họa việc sử dụng SVD để nén ảnh. Ảnh ban đầu kích thước *m×n*, có thể sử dụng biến đổi SVD cho ảnh kích thước *m×n* ở trên ta thu được 3 thành phần , Σ, , phần trắng của ma trận tượng trưng cho các phần tử không cần lưu trữ.

=

Amn

Umm

∑mn

VnnT

Hình1. Mô tả cách thức SVD nén ảnh.

Khi  đủ nhỏ, ta sẽ có tổng số cần lưu là  và nén ảnh bằng SVD trở nên hiệu quả. Chú ý rằng nếu ảnh A có hạng *r* nhỏ khiến cho  thì việc phân tích SVD sẽ làm giảm không gian lưu trữ mà không làm mất thông tin ảnh.

Sử dụng phần mềm Matlab, thực hiện nén ảnh sử dụng biến đổi SVD với các giá trị  khác nhau cho một tấm ảnh kích thước 177 × 284, kết quả thu được thể hiện ở Hình 2.

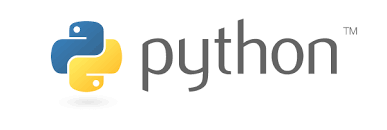
|  |  |
| --- | --- |
| Description: SVD_original  Hình gốc | Description: SVD_k_5  = 5 |
| Description: SVD_k_25  = 25 | Description: SVD_k_85  = 85 |

Hình 2. Ảnh gốc (a) và kết quả giải nén với các giá trị  khác nhau.

Đồng thời chúng tôi cũng đã vẽ được đồ thị mô tả sự sai số giữa ảnh gốc ban đầu và ảnh sau khi nén, cùng với đánh giá về hiệu quả nén khi áp dụng SVD với các giá trị  khác nhau, thể hiện ở hình trên

## 2.3 Giới thiệu về sử dụng python:

Python là một ngôn ngữ lập trình thông dịch (interpreted), hướng đối tượng (object-oriented), và là một ngôn ngữ bậc cao (high-level) ngữ nghĩa động (dynamic semantics). Python hỗ trợ các module và gói (packages), khuyến khích chương trình module hóa và tái sử dụng mã. Trình thông dịch Python và thư viện chuẩn mở rộng có sẵn dưới dạng mã nguồn hoặc dạng nhị phân miễn phí cho tất cả các nền tảng chính và có thể được phân phối tự do.



Sau đây là các đặc điểm của Python:

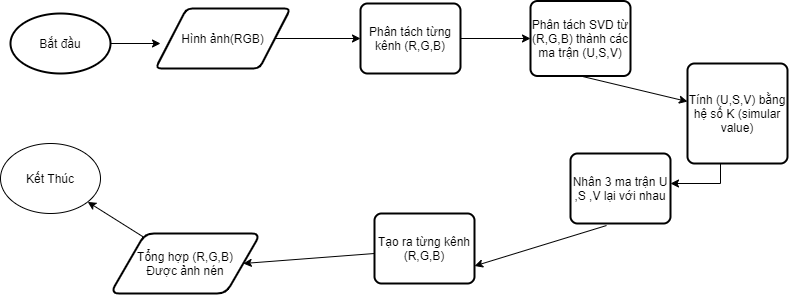
* Ngữ pháp đơn giản, dễ đọc.
* Vừa hướng thủ tục (procedural-oriented), vừa hướng đối tượng (object-oriented)
* Hỗ trợ module và hỗ trợ gói (package)
* Xử lý lỗi bằng ngoại lệ (Exception)
* Kiểu dữ liệu động ở mức cao.
* Có các bộ thư viện chuẩn và các module ngoài, đáp ứng tất cả các nhu cầu lập trình.
* Có khả năng tương tác với các module khác viết trên C/C++ (Hoặc Java cho Jython, hoặc .Net cho IronPython).
* Có thể nhúng vào ứng dụng như một giao tiếp kịch bản (scripting interface).
* Python dễ dàng kết nối với các thành phần khác:
* Python có thể kết nối với các đối tượng COM, .NET (Ironpython, Python for .net), và CORBA, Java… Python cũng được hỗ trợ bởi Internet Communications Engine (ICE) và nhiều công nghệ kết nối khác.
* Có thể viết các thư viện trên C/C++ để nhúng vào Python và ngược lại.
* Python là ngôn ngữ có khả năng chạy trên nhiều nền tảng.
* Python có cho mọi hệ điều hành: Windows, Linux/Unix, OS/2, Mac, Amiga, và những hệ điều hành khác. Thậm chí có cả những phiên bản chạy trên .NET, máy ảo Java, và điện thoại di động (Nokia Series 60). Với cùng một mã nguồn sẽ chạy giống nhau trên mọi nền tảng.
* Python rất đơn giản và dễ học
* Python có cộng đồng lập trình rất lớn, hệ thống thư viện chuẩn, và cả các thư viện mã nguồn mở được chia sẻ trên mạng.
* Python là ngôn ngữ mã nguồn mở

Cài đặt Python dùng giấy phép nguồn mở nên được sử dụng và phân tối tự do, ngay cả trong việc thương mại. Giấy phép Python được quản lý bởi Python Software Foundation.

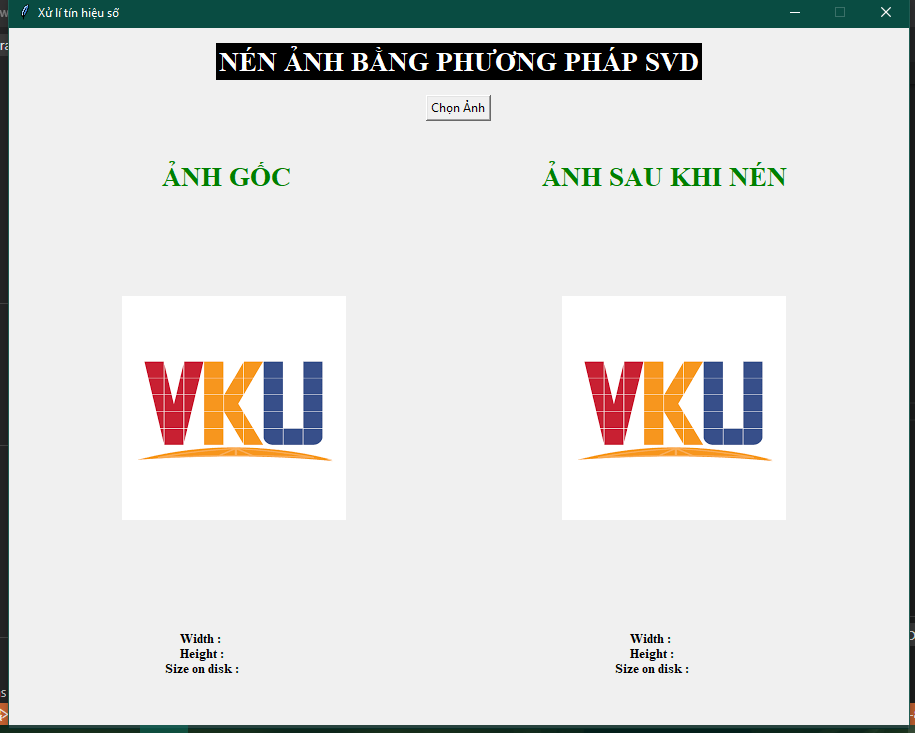
# CHƯƠNG 3: KẾT QUẢ MÔ PHỎNG BÀI TẬP

## 3.1 Phần mềm mô phỏng:

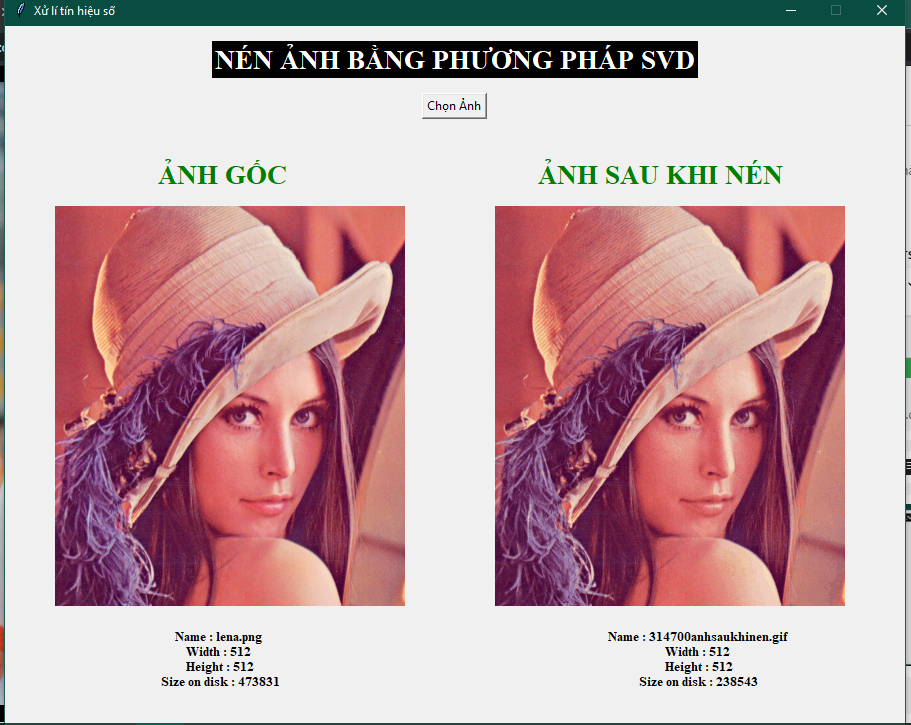
Lưu đồ thuật toán :



Chương trình :



Sau khi nén ảnh :



## 3.2 Kết quả

Ưu điểm: Đơn giản hóa dữ liệu, loại bỏ nhiễu, có thể cải thiện kết quả thuật toán

Nhược điểm: Dữ liệu được chuyển đổi có thể khó hiểu

## 3.3 Source Code :

import tkinter as tk

from tkinter.constants import \*

import os

from numpy import pad

from tkinter import Grid, PhotoImage, filedialog

from PIL import Image , ImageTk

import numpy

import random

def openImage(imagePath):

    imOrig = Image.open(imagePath)

    im = numpy.array(imOrig)

    aRed = im[:, :, 0]

    aGreen = im[:, :, 1]

    aBlue = im[:, :, 2]

    return [aRed, aGreen, aBlue, imOrig]

def compressSingleChannel(channelDataMatrix, singularValuesLimit):

    uChannel, sChannel, vhChannel = numpy.linalg.svd(channelDataMatrix)

   # aChannelCompressed = numpy.zeros((channelDataMatrix.shape[0], channelDataMatrix.shape[1]))

    k = singularValuesLimit

    leftSide = numpy.matmul(uChannel[:, 0:k], numpy.diag(sChannel)[0:k, 0:k])

    aChannelCompressedInner = numpy.matmul(leftSide, vhChannel[0:k, :])

    aChannelCompressed = aChannelCompressedInner.astype('uint8')

    return aChannelCompressed

def chonanh():

    pathname = filedialog.askopenfilename(initialdir = "E:\Xử lí ảnh\baitap",

                                          title = "Vui lòng chọn ảnh",

                                          filetypes = (("Image files",

                                                        "\*.png .jpg .jpeg"),

                                                       ("all files",

                                                        "\*.\*")))

    filename = os.path.basename(pathname)

    chonanhh(filename)

    aRed, aGreen, aBlue, originalImage = openImage(filename)

    singularValuesLimit = 431

    aRedCompressed = compressSingleChannel(aRed, singularValuesLimit)

    aGreenCompressed = compressSingleChannel(aGreen, singularValuesLimit)

    aBlueCompressed = compressSingleChannel(aBlue, singularValuesLimit)

    imr = Image.fromarray(aRedCompressed, mode=None)

    img = Image.fromarray(aGreenCompressed, mode=None)

    imb = Image.fromarray(aBlueCompressed, mode=None)

    newImage = Image.merge("RGB", (imr, img, imb))

    radomimg = random.randint(1, 1000000)

    newImage.save(str(radomimg)+"anhsaukhinen.gif")

    taolao = tk.PhotoImage(file=str(radomimg)+"anhsaukhinen.gif")

    test1.configure(image=taolao)

    test1.image = taolao

    infor1(filename)

    infor2(str(radomimg)+"anhsaukhinen.gif")

def chonanhh(filename):

    alo = tk.PhotoImage(file=filename)

    test.configure(image=alo)

    test.image = alo

def infor1(filename):

    image = Image.open(filename)

    width, height = image.size

    file\_stats = os.stat(filename)

    imform1.configure(text=f"Name : {filename} \nWidth : {width} \n Height : {height} \n Size on disk : {file\_stats.st\_size} ")

def infor2 (newname):

    image = Image.open(newname)

    width, height = image.size

    file\_stats = os.stat(newname)

    imform2.configure(text=f"Name : {newname} \nWidth : {width} \n Height : {height} \n Size on disk : {file\_stats.st\_size} ")

win = tk.Tk()

win.title("Xử lí tín hiệu số")

win.geometry('900x700')

win.resizable(width=False,height=False)

tieude = tk.Label(win,text="NÉN ẢNH BẰNG PHƯƠNG PHÁP SVD",font="Times 20 bold",fg="white",bg="black")

tieude.pack(side=TOP,pady=15)

FrameButton = tk.Frame(win)

FrameButton.pack(padx=120)

photo =tk.PhotoImage(file="logo.png")

test = tk.Label(win,image=photo)

test.place(width=350,height=400,x=50,y=180)

test1 = tk.Label(win,image=photo,width=200,height=200)

test1.place(width=350,height=400,x=490,y=180)

label = tk.Label(win,text="ẢNH GỐC",font="Times 20 bold",fg="green")

label.place(x =150,y = 130)

label2 = tk.Label(win,text="ẢNH SAU KHI NÉN",font="Times 20 bold",fg="green")

label2.place(x =530,y = 130)

chonAnh =  tk.Button(FrameButton,text="Chọn Ảnh",command=chonanh)

chonAnh.grid(row=1,column=1)

imform1 = tk.Label(text="Width : \n Height : \n Size on disk : ",font="Times 10 bold",fg="black")

imform1.place(x = 150,y = 600)

imform2 = tk.Label(text="Width : \n Height : \n Size on disk : ",font="Times 10 bold",fg="black")

imform2.place(x = 600,y = 600)

"""nenanh = tk.Button(FrameButton,text="Nén")

nenanh.grid(row=1,column=2) """

"""luuanh = tk.Button(FrameButton,text="Lưu Ảnh")

luuanh.grid(row=1,column=3)"""

win.bind("<Return>", chonanhh)

win.mainloop()