Middle East Research Journal of Engineering and Technology
ISSN: 2789-7737 (Print) ISSN: 2958-2059 (Online)
Frequency: Bi-Monthly
DOI: https://doi.org/10.36348/merjet.2026.v06i02.001

KUWAIT SCHOLARS PUBLISHER
Website: http://www.kspublisher.com/
Email: office@kspublisher.com

Matrix Operations and Illustrations in Graphics Engineering

Nguyễn Hoàng Vân[1*]
[1]Hạ Long University

Abstract: Linear algebra—particularly matrix theory—constitutes a foundational mathematical pillar in the domains of computer graphics and image processing. This paper establishes a linkage between fundamental matrix operations and image processing techniques, and presents a comprehensive analysis of their applications in the construction of virtual environments, spanning from elementary geometric transformations to sophisticated physics-based simulation algorithms. The objective of this study is to provide students with an intuitive yet practically grounded perspective, thereby narrowing the divide between abstract mathematical theory and real-world engineering applications. Furthermore, it serves as a pedagogical resource for instructors, enabling them to elucidate matrix operations through illustrative image processing examples in the classroom.

Keywords: Linear algebra, matrices, matrix operations, computer graphics.

Review Paper

***Corresponding Author:**
Nguyễn Hoàng Vân
Hạ Long University

How to cite this paper:
Nguyễn Hoàng Vân (2026). Matrix Operations and Illustrations in Graphics Engineering. *Middle East Res J. Eng. Technol, 6*(2): 34-39.

Article History:
| Submit: 17.03.2026 |
| Accepted: 15.04.2026 |
| Published: 18.04.2026 |

1. INTRODUCTION

Mathematics education in Vietnam is confronting new demands driven by the rapid advancement of science and technology, as well as the increasingly sophisticated needs of society. To effectively implement Resolution No. 29-NQ/TW (2013) of the Party Central Committee on fundamental and comprehensive reform of education and training—alongside Conclusion No. 91-KL/TW (2024) of the Politburo on its continued execution—integrating mathematics instruction with real-world contexts and applying mathematical knowledge to practical problem-solving is not merely a pedagogical approach, but a strategic imperative.

In recent years, Vietnam's mathematics curriculum has undergone notable improvements. Nevertheless, the tendency for students to engage in rote learning—without perceiving the connection between abstract theoretical knowledge and its practical applications—remains widespread. This disconnect often results in apprehension, or even aversion, toward mathematics among a segment of learners, thereby diminishing their interest and motivation.

Conclusion No. 91-KL/TW (2024) further underscores the importance of pedagogical innovation, emphasizing the development of learners' competencies and qualities rather than the mere transmission of knowledge. In the context of mathematics education, this entails a shift from teaching *what* to learn toward *how* to apply knowledge and *why* it matters.

Linear algebra is one of the most fundamental pillars of modern mathematics, exerting profound influence across numerous domains of science, technology, and economics. In particular, matrices represent a central concept within linear algebra, playing a pivotal role in addressing a wide array of complex problems. However, many widely used linear algebra textbooks remain predominantly theoretical, prioritizing conciseness and conceptual clarity while offering limited engagement with real-world applications of matrices.

Within the broader trajectory of modern mathematics, matrices have evolved beyond a mere algebraic tool for solving systems of linear equations. They have become a universal "language," forming the backbone of data science and computer engineering. Drawing on years of teaching experience and with the aim of enhancing instructional quality, this study proposes a set of practical problems for teaching matrix multiplication, matrix transposition, and determinant computation in linear algebra courses. These topics are explicitly connected to applications in computer graphics and image processing.

It is hoped that these approaches will support mathematics educators in designing context-rich problems, thereby enriching instructional content and fostering greater creativity and pedagogical effectiveness in the teaching of mathematics.

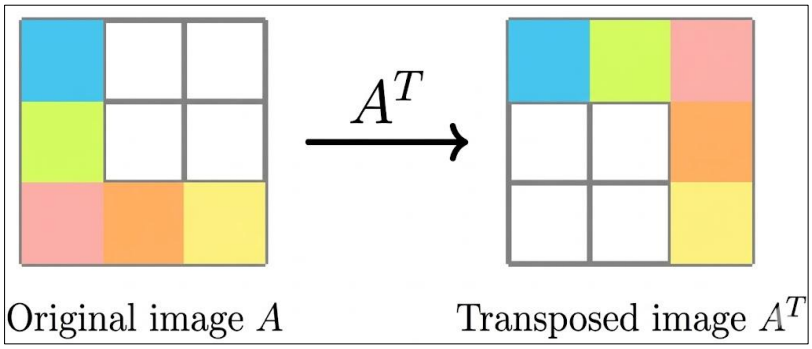
2. Applications of Selected Matrix Operations in Computer Graphics and Image Processing

2.1. Application of Matrix Transposition

For any matrix $A = (a_{ij})_{m\times n}$, , its transpose is defined as the matrix obtained by interchanging rows and columns, commonly denoted by A^T. In digital image processing, an image can essentially be represented as a matrix of pixels. When teaching matrix transposition, instructors may pose a simple yet thought-provoking question such as: Which image transformation techniques correspond to the operation of interchanging rows and columns?

A number of high-achieving students with strong spatial reasoning skills may arrive at the correct answer immediately. However, most students may find this question challenging. In such cases, instructors can provide guided multiple-choice hints—for example: a 90-degree clockwise rotation, a 90-degree counterclockwise rotation, image flipping, or reflection across the main diagonal—to help orient students' thinking.

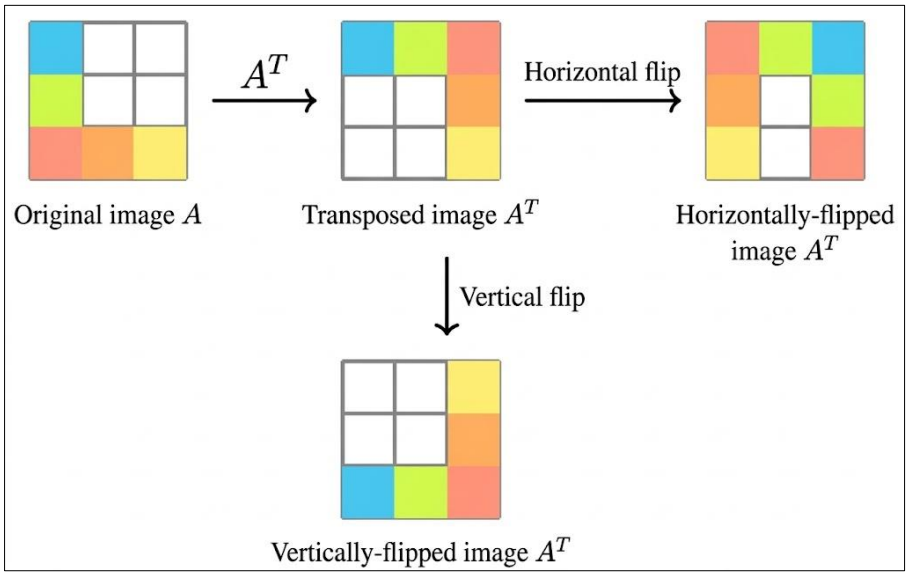
The correct answer is reflection across the main diagonal. If an image is regarded as a square matrix, the transpose operation effectively reflects the image along the diagonal running from the top-left corner to the bottom-right corner, as illustrated in the figure below.



In fact, matrix transposition alone does not produce a complete 90° rotation; rather, it serves as the most critical intermediate step. In practice, to rotate an image by 90°, graphics libraries typically perform a combination of operations:

- **90° clockwise rotation**: Transpose the matrix → then apply a horizontal flip.
- **90° counterclockwise rotation**: Transpose the matrix → then apply a vertical flip.

This decomposition highlights how fundamental matrix operations underpin common image transformation techniques in computer graphics.



When explaining this illustration, instructors can guide students to use colors to trace individual pixels through each transformation step.

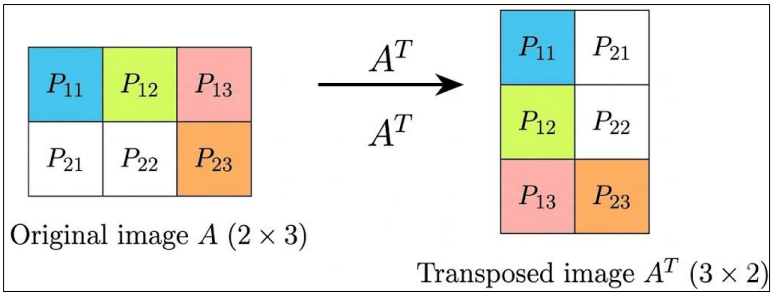
In Step 1(A→A^T), the entire first row becomes the first column; similarly, the second row becomes the second column, and the third row becomes

the third column. The elements a_{11}, a_{22}, a_{33} remain fixed along the main diagonal.

In Step 2 **(**A^T**)**, flipping the transposed matrix horizontally transforms column 1 into column 3, leaves column 2 unchanged, and maps column 3 to column 1. If we compare the final image with the original by mentally rotating the original image 90° clockwise, we observe that the positions of the colored pixels align perfectly.

When the image is rectangular rather than square, the transpose A^T still rigorously follows the mathematical rule $a_{ij} \rightarrow a_{ji}$, yet the visual effect introduces an interesting "reshaping" of the frame. First, the dimensions are completely swapped: an m×n image becomes n×m, turning a landscape image into portrait orientation, and vice versa. Second, despite this change in dimensions, the only fixed "anchor point" is the top-left corner (1;1) here, the main diagonal acts as a conceptual hinge, keeping pixels with coordinates i=j in place while all other elements are reflected across this implicit axis.

Finally, it is important to emphasize that transposition is not merely a 90° rotation, but rather a combination of reflection and reorientation. It effectively reverses the orientation of objects—for instance, a person appearing to use their right hand in the original image may appear left-handed and tilted in the transformed image.



2.2. Applications of Matrix Multiplication

Matrix multiplication can be viewed as a natural extension of real-number multiplication; however, it is far more than a simple row-by-column operation—it represents the composition of linear transformations. In robotics, a fundamental objective is to determine the position of a robot's end-effector (e.g., its hand) based on the rotation angles of its joints. A robotic arm typically consists of multiple sequential joints (such as the shoulder, elbow, and wrist), and the final position of the end-effector relative to a reference coordinate system is obtained by successively multiplying the transformation matrices corresponding to each joint.

Similarly, in computer graphics, basic geometric operations—such as rotation, translation, and scaling—are systematically reduced to matrix multiplications.

2.2.1. Scaling Transformation

In two-dimensional space, resizing an image—either enlarging or shrinking—is achieved by multiplying the coordinates (x,y) of each point by a 2×2 diagonal matrix. At this stage, instructors may pause to pose an engaging question:

> An image of arbitrary size—for instance, a Full HD image with a resolution of 1920×1080, comprising over two million pixels—can be viewed as a large matrix of color values. If we only use a 2×2 matrix, can matrix multiplication still effectively scale the image?

The answer reveals a subtle yet fascinating concept that often confuses beginners in computer graphics. The key lies in distinguishing between the coordinates of a point and the number of points. A transformation matrix is fundamentally different from an image data matrix. The latter stores color information, whereas the former acts as a set of mathematical rules governing how points are transformed. It does not contain color values; instead, it encodes how positions change.

In two-dimensional space, each point is defined by two components (x,y). To transform (x,y) into (x′,y′), we require a matrix capable of operating on these two components—hence the use of a 2×2 matrix for scaling. The scaling matrix SSS is defined as:

$$S = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix},$$

Here s_x and s_y denote the scaling factors along the x-axis and y-axis,, respectively. When matrix multiplication is applied to a point P(x,y), the coordinates of the transformed point P′(x′,y′) are obtained as follows:

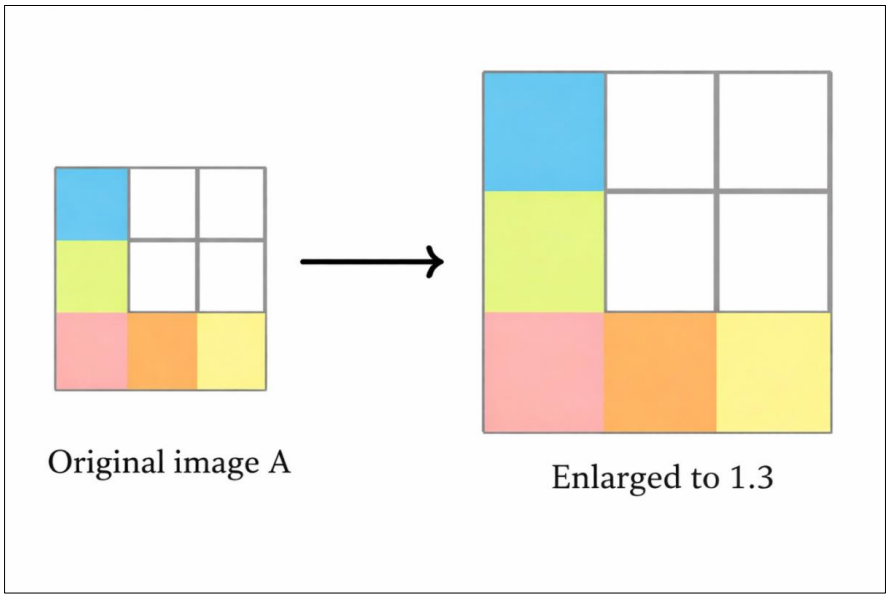
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} s_x \cdot x \\ s_y \cdot y \end{bmatrix}.$$

This formulation clearly demonstrates how scaling transformations independently adjust each coordinate component.

Typical scaling cases:

(i) Uniform scaling**:** When $s_x = s_y$, the object preserves its original aspect ratio while only its overall size changes (either enlarging or shrinking).

For instance, if the original image is enlarged by a factor of 1.3 as illustrated below, instructors may prompt students to predict the corresponding scaling matrix used in this transformation.



The resulting scaling matrix in this case is:

$$S = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} = \begin{bmatrix} 1.3 & 0 \\ 0 & 1.3 \end{bmatrix}.$$

(ii) Non-uniform scaling: When $s_x \neq s_y$. the object is stretched or compressed, altering its original aspect ratio. For example $s_x = 2$ và $s_y = 1.3$, the object not only enlarges overall but also experiences greater horizontal stretching than vertical, resulting in a distortion of the original aspect ratio.

(iii) Reflection (Negative Scaling): If $s_x = -1, s_y = 1$: The object is flipped horizontally across the **y-axis**, which corresponds to the horizontal flip illustrated in the 90°

If $s_x = 1, s_y = -1$: The object is flipped vertically across the **x-axis**, corresponding to the vertical flip shown in the 90° counterclockwise rotation example above.

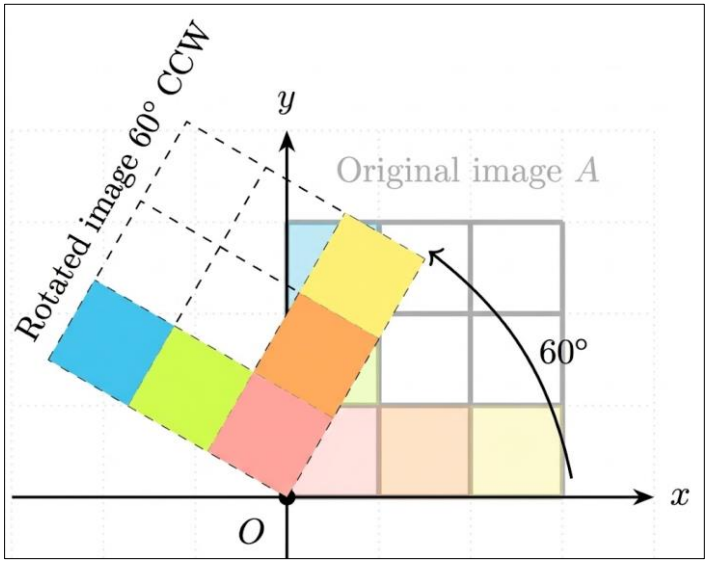
2.2.2. Rotation

To rotate a point $P(x, y)$ by an angle θ counterclockwise around the origin, we use the rotation matrix R:

$$R = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}.$$

Applying this matrix to the point $P(x, y)$, the coordinates of the rotated point $P'(x', y')$ are computed as:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x\cos\theta - y\sin\theta \\ x\sin\theta + y\cos\theta \end{bmatrix}.$$



Question: *Determine the rotation matrix for a 60° rotation illustrated in the figure above.*

Answer: *The counterclockwise rotation matrix for 60∘ is given by:*

$$R = \begin{bmatrix} \cos 60^0 & -\sin 60^0 \\ \sin 60^0 & \cos 60^0 \end{bmatrix} \approx \begin{bmatrix} 0.5 & -0.866 \\ 0.866 & 0.5 \end{bmatrix}.$$

2.2.3. Translation

For scaling or rotation, we can use 2×2 matrix multiplication because these are linear transformations that always fix the origin (0,0). However, translation is an affine transformation. Its essence lies in vector addition:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix},$$

Here t_x và t_y These correspond to the displacements along the respective axes Ox và Oy. From an algebraic perspective, no 2×2 matrix can be multiplied by $[x\ y]^T$ to achieve the addition of a constant vector. This limitation creates a "break" in the Graphics Pipeline, because the GPU architecture prefers to consolidate all transformations—translation, rotation, and scaling—into a single chain of matrix multiplications for efficiency.

To address this issue, mathematicians and graphics engineers use a "trick": they lift the 2D space onto a virtual plane in 3D by adding a homogeneous coordinate $z = 1$. At this point, the point $P(x, y)$ becomes $P(x, y, 1)$. The translation can then be represented as a 3×3 matrix:

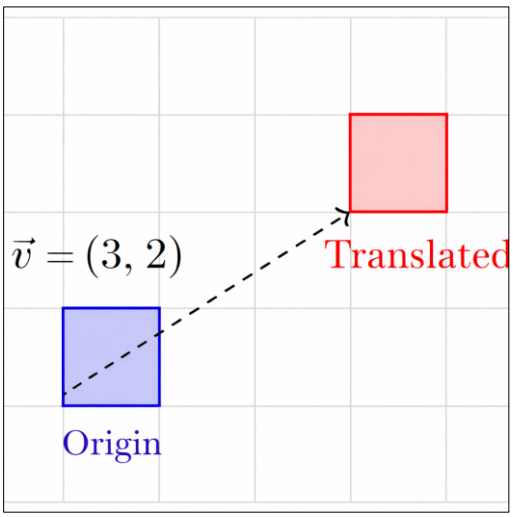
$$T = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}.$$

When performing matrix multiplication:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} x + t_x \\ y + t_y \\ 1 \end{bmatrix}.$$

By inserting t_x và t_y into the third column, matrix multiplication effectively "absorbs" the addition operation. Once the computation is complete, the computer simply discards the final 1 to return to the original 2D coordinates.

Question: *Determine the translation matrix T illustrated in the figure below.*



Answer: the corresponding translation matrix T is:

$$T = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 2 \\ 0 & 1 & 3 \\ 0 & 0 & 1 \end{bmatrix}.$$

2.2.4. Composite Matrix

Suppose we want to perform the following sequence of transformations on an object: scale the object, rotate it around the origin, and then translate it to a new position.

In mathematical terms, if v is the original coordinate vector, the new coordinates v′ are computed as:

$$v' = T \cdot \big(R \cdot (S \cdot v)\big).$$

so that the transformed vector is simply:

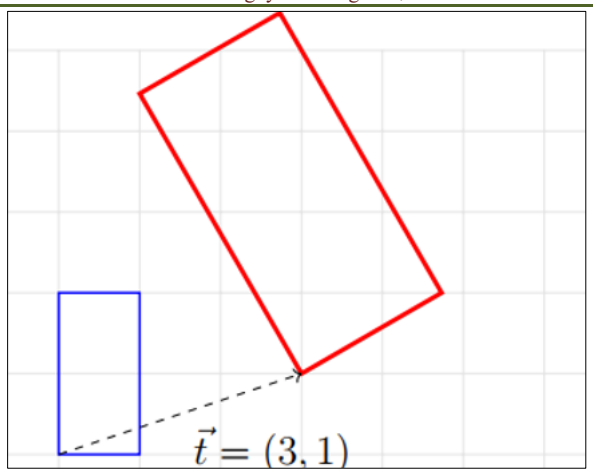
$$v' = (T \cdot R \cdot S) \cdot v \Rightarrow v' = M \cdot v,$$

Here, $M = T \cdot R \cdot S$ is the composite transformation matrix.

To perform the combined multiplication, we first upgrade the S và R matrices to 3×3. The resulting general form of a 3×33 \times 33×3 affine transformation matrix is:

$$M = \begin{bmatrix} s_x \cos\theta & -s_y \sin\theta & t_x \\ s_x \sin\theta & s_y \cos\theta & t_y \\ 0 & 0 & 1 \end{bmatrix}.$$

Here is an example of a combined transformation: scale by 2, rotate 30° counterclockwise, and translate by (3, 1).



2.3. Determinant

The determinant is not just an abstract number; it represents the scale factor of space, measuring how much a transformation stretches or compresses the space.

If a transformation matrix A has det(A)=k, then the area in 2D (or volume in higher dimensions) is scaled by |k|| after the transformation. This provides a very intuitive way to visualize the otherwise abstract determinant as a concrete geometric quantity.

For example, consider a unit square with an area of 1. After applying the matrix transformation A, the square becomes a parallelogram. The area of this new parallelogram is exactly |det (A)|:

General formula:
New area=|det(A)|×Original area.
If we scale an object by 2 along the xxx-axis and 1.3 along the yyy-axis, the corresponding scaling matrix is:

$$S = \begin{bmatrix} 2 & 0 \\ 0 & 1.3 \end{bmatrix} \Rightarrow \det(S) = 2.6.$$

This means that any shape or image undergoing this transformation will have its area increased by a factor of 2.6 compared to its original size.

A pure rotation and translation do not change the size of an object.
The 2D rotation matrix for a counterclockwise rotation by an angle θ\thetaθ is:

$$R = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \Rightarrow \det(R) = \cos^2\theta + \sin^2\theta = 1.$$

Since the determinant of a rotation matrix is always 1, the area of the object is perfectly preserved. Similarly, a translation can be represented as a 3×3 matrix in homogeneous coordinates:

$$T = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \Rightarrow \det(T) = 1.$$

Since the determinant is always 1, the area of the object is completely preserved.

3. CONCLUSION

This paper presents a fresh perspective on matrix transposition, matrix multiplication, and determinants. Linear algebra is far more than abstract numbers; it serves as a fundamental backbone of computer graphics, participating in every stage—from plotting a single point on the screen to simulating complex lighting and physical effects.

The paper systematizes basic matrix operations and provides a detailed analysis of their applications in constructing virtual graphics, ranging from basic geometric transformations to complex physics-based simulation algorithms.

Based on the content of this paper, instructors gain a new way to explain the meaning of matrix operations in the classroom. Simultaneously, it offers students a visual and practical understanding, helping to bridge the gap between pure mathematical theory and real-world engineering applications.

REFERENCES

- Đồng Văn Việt (2025), *Creating Practical Problems to Illustrate Matrix Operations*, *Teaching and Learning Today Journal*, September 2025 issue.
- Howard Anton, Chris Rorres (2014), *Elementary Linear Algebra, Applications Version*, Wiley, America.
- Nguyễn Đình Chí, Tạ Văn Đĩnh, Nguyễn Hồ Quỳnh (2006), *Advanced Mathematics – Volume 1*, Education Publishing House.
- Rafael C. Gonzales, Richard E. Woods (2022), *Digital Image Processing*, Addison-Wesley.
- Steven J. Leon (2015), *Linear Algebra with Applications*, Pearson, America.