

# **SECV1113-01**

# **MATHEMATICS**

# **FOR COMPUTER GRAPHICS**

GROUP PROJECT

“COVID 19 - STAY AT HOME”

## LECTURER: SURIATI BINTI SADIMON

|  |  |  |
| --- | --- | --- |
| **NO.** | **GROUP MEMBER** | **MATRIC NO.** |
| 1 | MOHD ANAS BIN ADNAN (Leader)  contact:0187804722 | B19EC0043 |
| 2 | ROSHANDEV DANIEL | A19EC0156 |
| 3 | NUR HASANAH BINTI SARIDDON | B19EC0033 |
| 4 | NURSYAFIQAH BINTI ABDUL HALIM | A19EC3025 |
| 5 | NG JING ER | A19EC0115 |

INTRODUCTION

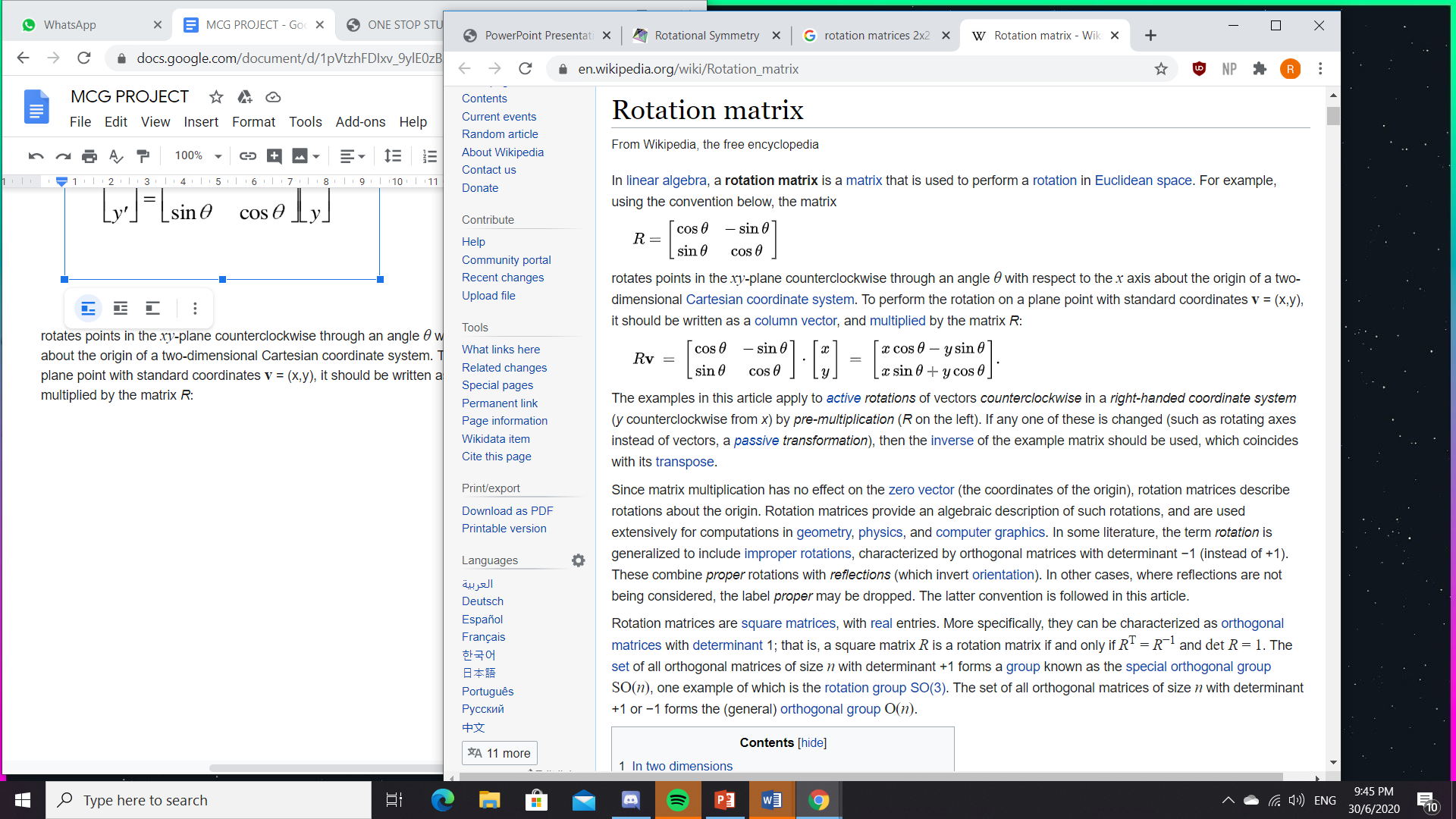
Based on the theme “COVID 19- stay at home” of this project for the course Mathematics For Computer Graphics, we have designed a simple graphic program using dev C++. Due to the restriction of movement order, we are not allowed to go back to our campus, therefore we have carried out all the discussions via WhatsApp instead of face to face discussion.

So, in this project we have implemented the graphic library to generate some basic shapes using the graphic functions such as putpixel(), circle(), line(), drawpoly() and so on. We also implemented a few functions based on the trigonometric interpolation and 2D transformations including rotation, translation and scaling to generate a desired graphic image for our project. In this project report, we will further explain the application of 4 functions for 2D transformation and trigonometric interpolation that have been used in our project.

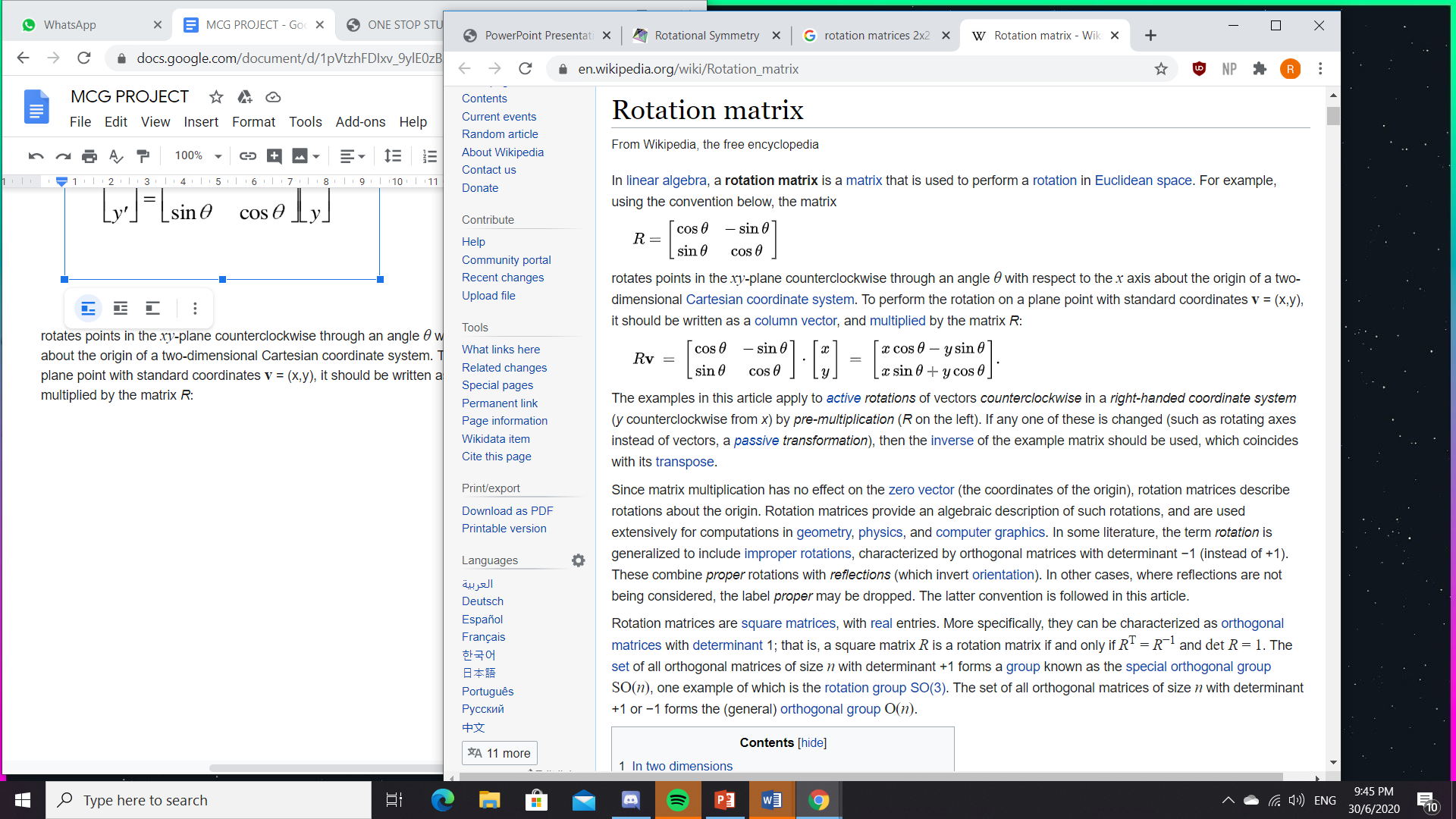
FUNCTIONS FOR 2D TRANSFORMATION AND INTERPOLATION

1. Rotation

A rotation repositions all points in an object along a circular path in the plane which is centered by something known as a pivot point. This way we can assume that the pivot point is actually the point of origin at where the rotation of an object or vector happens. In linear algebra, a **rotation matrix** is a matrix that is used to perform a rotation in Euclidean space. For example, using the convention below, the matrix



rotates points in the *xy*-plane counterclockwise through an angle *θ* with respect to the *x* axis about the origin of a two-dimensional Cartesian coordinate system. To perform the rotation on a plane point with standard coordinates **v** = (x,y), it should be written as a column vector, and multiplied by the matrix *R*:



p' x = px cos – py sin

p' y = px sin + py cos

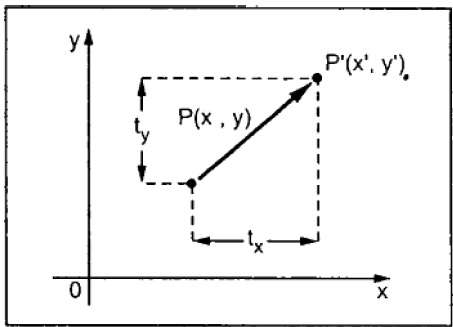
Or in matric form:

P’ = R **.** P

Based on our project, the rotation was done for the output of the house. The rotation was done for the line on the roof in which the line for the right roof originated from the line of the left roof and was done by using the method of rotation.

1. Translation

Translation is a movement of objects without deformation. Every position or point is translated by the same amount. When the straight line is translated, it will be drawn using endpoints. A point in 2D can be translated by adding translation coordinate (tx, ty) to the original coordinate (X,Y) to get the new coordinate (X’, Y’).



From the figure above, it is written that X’ = X + tx , Y’ = Y + ty. The pair (tx,ty) is called the translation vector or shift vector. The above equations can also be represented using column vectors.

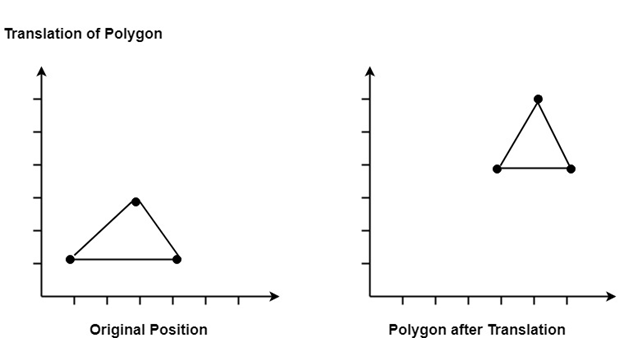


We can write as P’ = P + T.

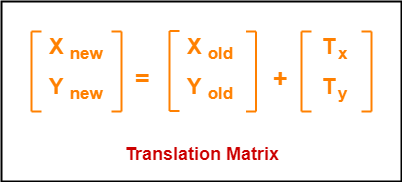
Based on our project, translation has been implemented on the house shape where a long vertical line was formed by translating the left vertical line. Translation also implemented on forming the heart rate line. The second heart rate formed by translating all the lines formed in the first heart rate.

Example of translation :

Translation of polygon



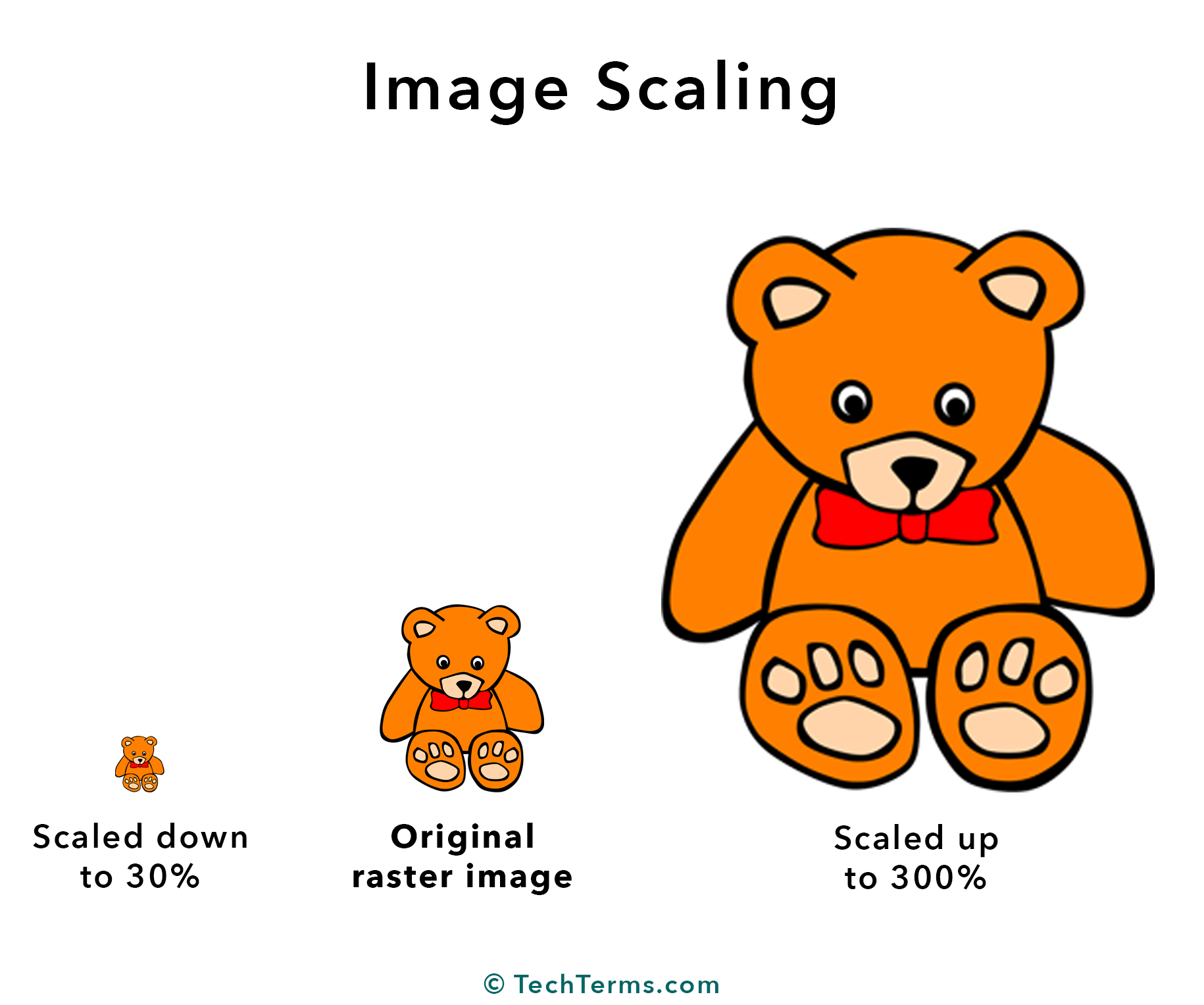
In matrix form, translation of equation may represented as : -



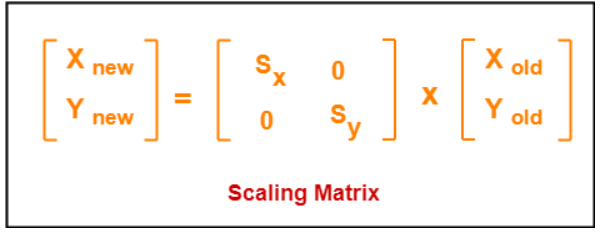
* The homogeneous coordinates representation of (X, Y) is (X, Y, 1).
* Through this representation, all the transformations can be performed using matrix / vector multiplications.

1. Scaling

Scaling is the process of modifying and altering the size of an object. Scaling also can be used to reduce the size of an object. The increase or reduce the object size is determined by the scaling factor. When the scaling factor is higher than 1, the object form is increased. When the scaling factor is below 1, the object form is reduced from the main object. Below is the example of scaling of an Object. The original image of the bear will be in the middle and at the right will be a scale of the bear with an increase of the size of 300%. At the left will be the object of image that shrinks with 30%. This shows the concept of the scaling in computer graphics.



To calculate the value of the scaling image we can use the matrix. The matrix will represent the value of coordinates. For example, the initial coordinate of the object O=(Xold,Yold). The scaling factor for X-axis = Sx. Scaling factor for Y-axis = Sy. New coordinate for image of object O will be (after scaling = (Xnew,Ynew)). This calculation also can be done by using Matrix. This scaling is achieved by scaling equation Xnew = Xold x Sx and Ynew = Yold x Sy. This will form the matrix equation as below.



In our coding for the final project, we have implemented the concept of scaling. The concept of scaling is implemented at the object of heart. The initial image of the heart is a small heart that is located inside the house. Using this coordinate image, we have put the scaling factor so that the image of the heart will form bigger than its initial image. We have used the class function of Scale to implement this concept. We also initialize the value of x, y, sx and sy using data type int. Value of x and y will be the coordinate of the initial image of the heart. For sx and sy, it will be the value of the scaling factor. The value of n is the value of a new object form. In class function Scale, we have implemented the matrix formula. This will make the system calculate the value of the scale.

We also use the class function of findNewCoordinate. This class function is to form the image away from the initial image. If we do not use this function the image form for scaling will overlap. It will be hard for us to see the concept of scaling. Class function findNewCoordinate also uses a matrix to find new coordinates. The concept of scaling is also implemented in this class function.

In conclusion, the concept of scaling is used in our coding for the final project of Mathematic Computer Graphics. Scaling concept is important in the way to form graphics images. Scaling concept is also used in 2D and 3D platforms. By learning this topic we can relate the concept of scaling in Mathematic Computer Graphics and the real world of Computer Graphics.

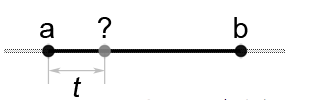
1. Interpolation

Interpolation could be a very commonly used technique in tricks. fairly often data is specified on an everyday grid (values are written at the vertex position of a 2D or 3D grid) or on a line (in the 1D case) but the program must evaluate values randomly position thereon grid. If the sample is found on a grid vertex, then we are able to simply use the worth that's stored there. But if the sample is found anywhere else on the grid (somewhere within the cell) then, considering that we've no data there, we'd like to compute one by averaging values which are stored at the cell vertices. This method is termed interpolation because the key idea is to "interpolate" existing values at fixed grid location to compute values anywhere else on the grid.

In 2D the technique is termed bilinear interpolation. Its 3D counterpart is termed trilinear interpolation. Both techniques are going to be described within the next two chapters and ASCII text files are given similarly. The word linear is in both terms because for that exact technique only linear interpolations are performed. A linear interpolation is an equation of the kind:

This is very just like the method of evaluating a linear function. This method is easy, requires only two values (a and b) and some simple arithmetic operations. Notice that t is within the range 0 to 1. However, as we are going to show within the next chapter, linear interpolation creates "visual" patterns which aren't always acceptable or desirable. it's possible to use interpolation methods of upper degrees which give smoother results (depending on the context, such results don't seem to be always considered better). to attain such interpolation though it's often necessarily to require into consideration quite the four cell corners surrounding a sample point. Therefore, they supply better results but at a better computation cost because they typically need a bigger set of points and use a function of degree two or more. The function accustomed to interpolate the values on the regular grid is termed the interpolant.

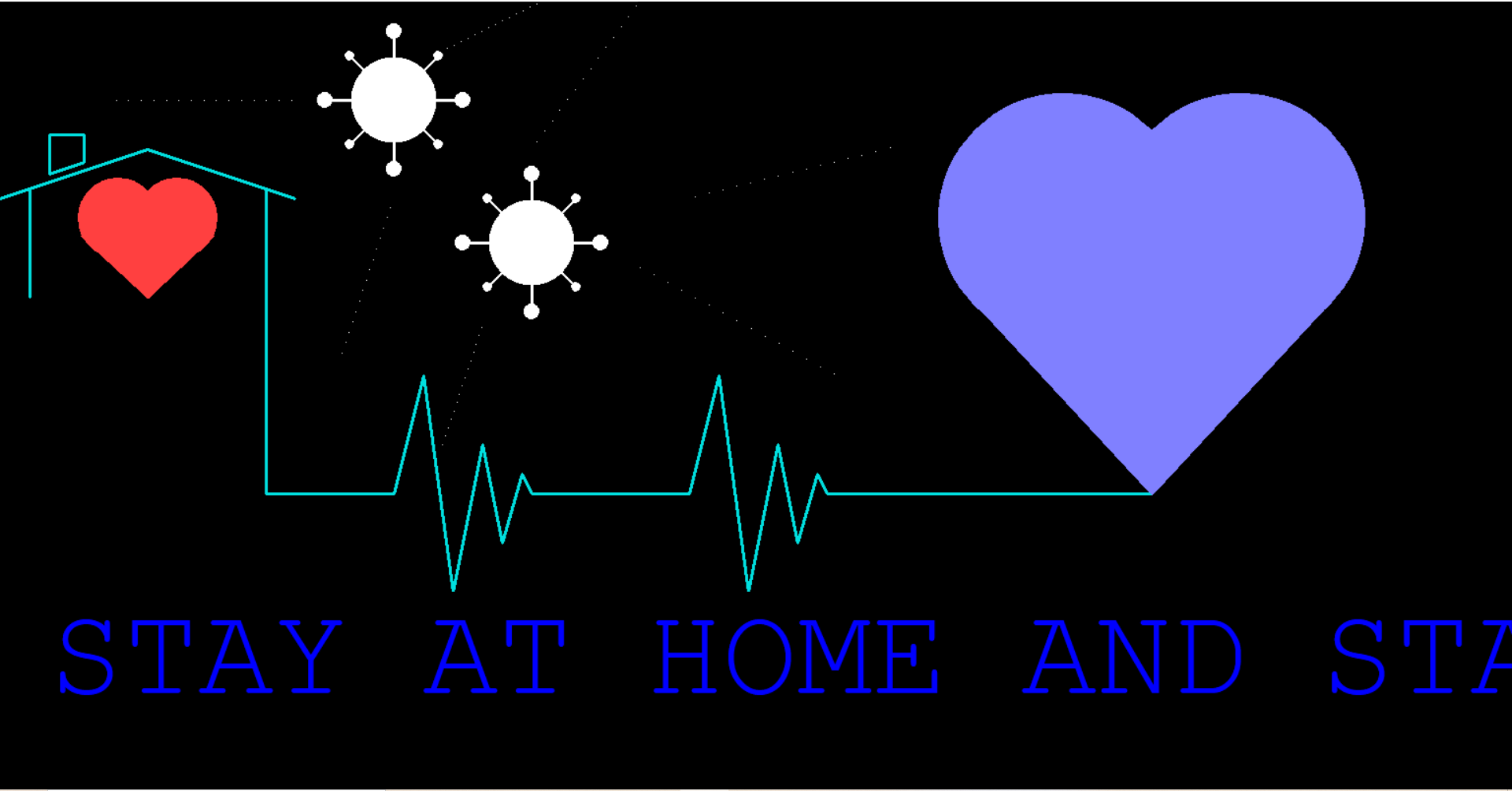
Interpolation techniques are commonly used in image processing (to resize images for instance). But 3D techniques involve the utilization of 3D or 2D grids (textures are seen as 2D grids) like for example fluid simulation, volume rendering, texture mapping and irradiance caching just to call some. Wherever grids are involved, interpolation techniques are usually also needed.



We have designed several shapes using functions based on 2D transformation and interpolation as shown in the table below:

|  |  |  |
| --- | --- | --- |
| Shape | function | Example output |
| House | * rotation   Form the right roof of the house by rotating the left roof.   * translation   Form a long vertical line by translating the left vertical line. |  |
| Heart rate | * translation   Form the second “heart rate” by translating all the lines formed in the first “heart rate”. |  |
| Red heart | Form the shape of heart using 2 red filled circles and 1 polygon |  |
| Blue heart | * scale   Form a bigger blue heart by scaling both of the circles of red heart and change the colour to lightblue. |  |
| Virus | The shape of the virus didn’t make use of any 2D transformation and interpolation where it only used plenty of circle() functions and line() functions. But it will perform motion graphics instead of static graphics when executing the program. |  |
| Dotline | * interpolation   Form dot lines that surround the virus by using the trigonometric interpolation function. In the interpolation function, dots are generated with increasing spacing between the interpolated values that are set as the parameter.    (The dotlines are the “pathways” of virus spreads.) | \*You may need to adjust the brightness to see a clearer image |
| Message | A message “STAY AT HOME AND STAY SAFE!” is displayed using the function printMsg(). | \*The message is displayed in motion and the colour of the message will keep changing. |

EXAMPLE OF OUTPUT



MEMBER COMMITMENT

|  |  |  |
| --- | --- | --- |
| NO. | NAME | COMMITMENT |
| 1. | MOHD ANAS BIN ADNAN | C++ Program: Scaling function  Report : Explain scaling function applied in project |
| 2. | ROSHANDEV DANIEL | C++ Program: Rotation function  Report : Explain rotation function applied in project |
| 3. | NUR HASANAH BINTI SARIDDON | C++ Program: Interpolation function  Report : Explain interpolation function applied in project |
| 4. | NURSYAFIQAH BINTI ABDUL HALIM | C++ Program: Translation function  Report : Explain translation function applied in project |
| 5. | NG JING ER | C++ Program: Main function  Report : Generate a simple table to explain the output of the project. |

REFERENCES

1. Computer Graphics Translation - javatpoint.(2011) retrieved from <https://www.javatpoint.com/computer-graphics-translation>
2. Akshay Singhal. ( 2019. December 21) retrieved from <https://www.gatevidyalay.com/2d-transformation-in-computer-graphics-translation-examples/>
3. 2D Transformation. (2020) retrieved from <https://www.tutorialspoint.com/computer_graphics/2d_transformation.htm>