Đồ họa

Tuần 3

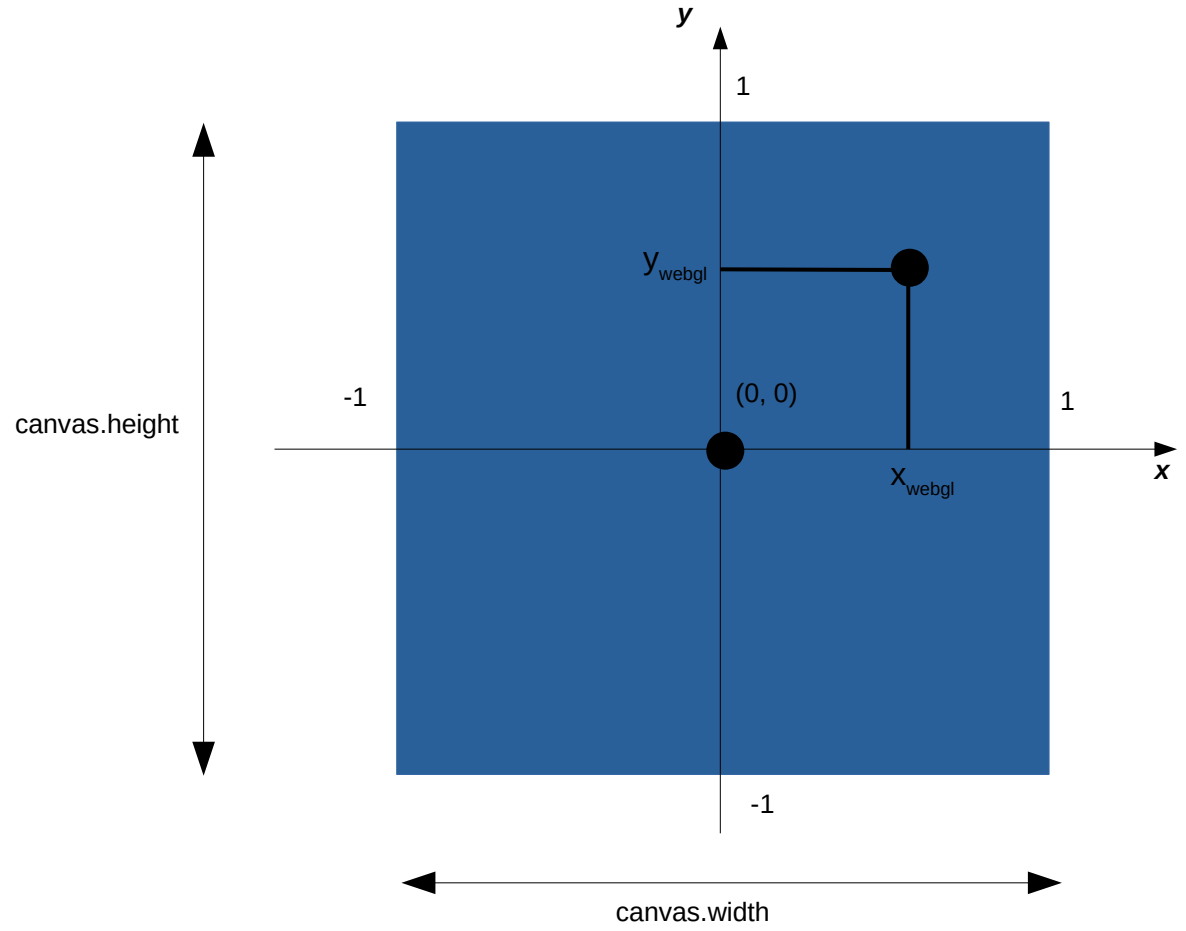
Giảng viên: Trần Đức Minh

Nội dung bài giảng

- Biến đổi tọa độ người dùng
- Khung nhìn
- Phép biến đổi Affine
- Phép biến đổi hình 2D
- Các ví dụ

Biến đổi tọa độ người dùng

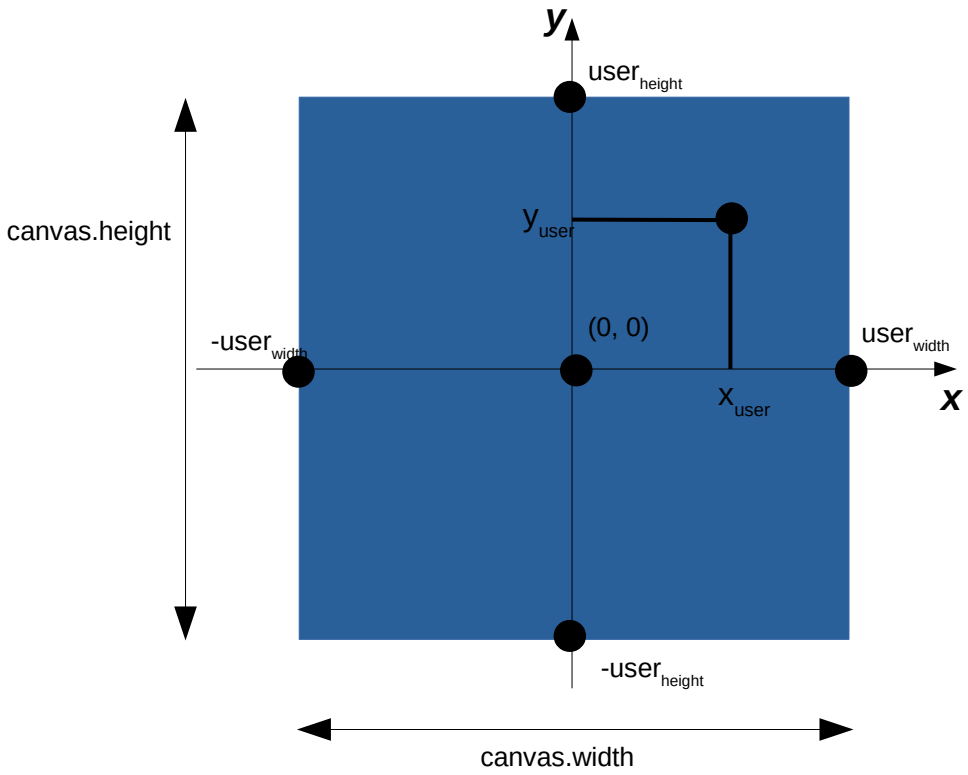
- Trục tọa độ 2D của WebGL



Biến đổi tọa độ

- Trục tọa độ 2D của người sử dụng
  - Khoảng giá trị
    - [-user$_{width}$, +user$_{width}$] đối với trục x
    - và [-user$_{height}$, +user$_{height}$] đối với trục y

  được dùng để xác định vùng tọa độ dành cho các thành phần đối tượng có thể thấy được trên màn hình.



Biến đổi tọa độ

- Trục tọa độ của người sử dụng
  - **Thuận tiện** hơn cho người sử dụng trong việc dựng hình và tính toán.
- Sau khi dựng hình cho đối tượng dựa trên trục tọa độ người sử dụng, cần **biến đổi** các tọa độ này từ ***trục tọa độ người sử dụng*** sang ***trục tọa độ WebGL*** để hệ thống có thể **vẽ đối tượng** lên màn hình.
- Công thức:
  - $x_{webgl} = x_{user} / user_{width}$
  - $y_{webgl} = y_{user} / user_{height}$

Biến đổi tọa độ

- Ví dụ:
  - Hệ trục tọa độ của người sử dụng được định nghĩa như sau
    - Trục x nằm trong khoảng [-640, 640] **($user_{width}$ = 640)**
    - Trục y nằm trong khoảng [-480, 480] **($user_{height}$ = 480)**
  - Chuyển đổi tọa độ điểm **(x_{user} = 120, y_{user} = 120)** trên hệ trục tọa độ của người sử dụng sang hệ trục tọa độ WebGL
    - $x_{webgl} = x_{user} / user_{width} = 120 / 640 = 0.1875$
    - $y_{webgl} = y_{user} / user_{height} = 120 / 480 = 0.25$

Hàm trong GLSL

- Hàm trong GLSL được định nghĩa giống như trong C/C++.
  - **Phải được khai báo** trước khi đem ra sử dụng (nguyên mẫu hàm) nếu phần triển khai hàm được đặt bên dưới hàm main.
  - **Không được phép gọi đệ quy**.
  - Khi **truyền đối số cho hàm là một mảng**, ta cần phải xác định rõ số lượng phần tử của mảng.
    - Ví dụ: **void tinhToan(vec3 values[20])**;

Ví dụ 1

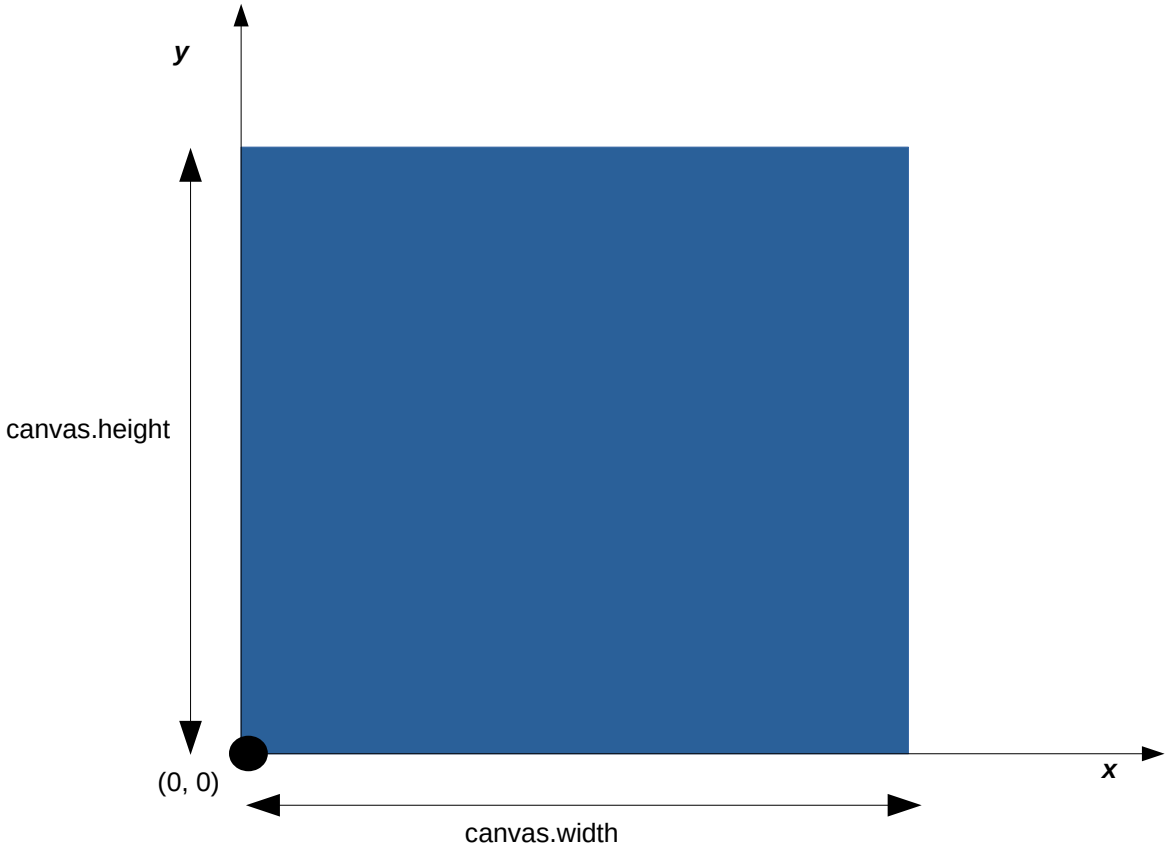
- Vẽ một hình tam giác dựa trên tọa độ người sử dụng.
  - Xây dựng một **hàm** để biến đổi từ tọa độ người sử dụng sang tọa độ WebGL trong GLSL.
  - Chạy thử nghiệm:
    - Thay đổi kích thước canvas và quan sát
      - Chiều rộng, chiều cao bằng nhau
      - Chiều rộng, chiều cao không bằng nhau

Ví dụ 2

- Vẽ trục tọa độ và một chữ T lên màn hình dựa trên tọa độ người sử dụng. Ví dụ này quan tâm đến cấu trúc xây dựng chương trình.
  - Tách riêng phần định nghĩa bộ đệm ra thành một hàm riêng biệt.
  - Xây dựng **2 hàm render**
    - Một hàm render dùng để vẽ trục tọa độ
    - Một hàm render dùng để vẽ chữ T
      - Vẽ chữ T bằng việc vẽ các hình tam giác nhỏ hơn thông qua hàm **setRectangle()**

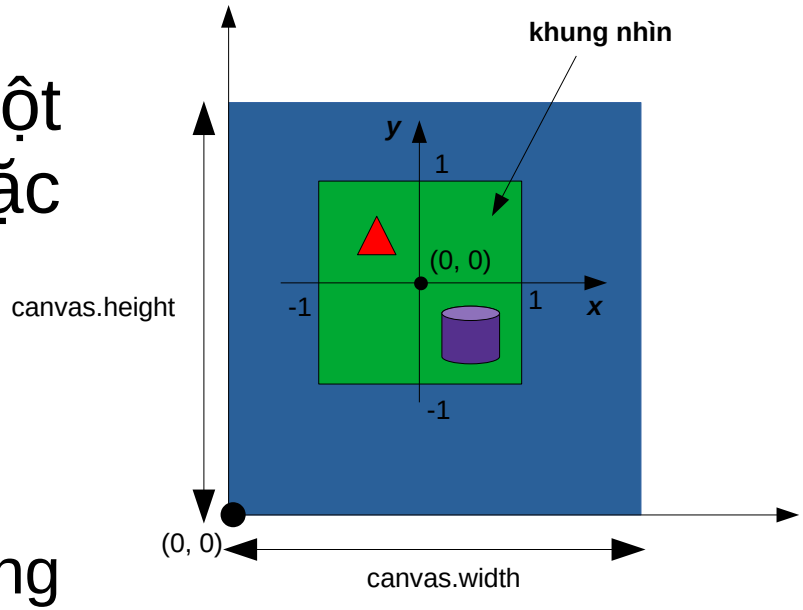
Trục tọa độ thế giới

- **Gốc tọa độ** nằm ở góc dưới trái của canvas



Khung nhìn

- **Khung nhìn** (**viewport**) là một khung cửa sổ bao gồm các đặc điểm sau
  - Được định nghĩa bởi người dùng.
  - Nằm bên trong canvas.
  - Dùng để chứa toàn bộ các đối tượng hình ảnh được vẽ trên hệ trục tọa độ WebGL.
- Canvas có thể **chứa nhiều khung nhìn** bên trong nó, nhưng ta chỉ có thể vẽ lên khung nhìn được định nghĩa sau cùng.

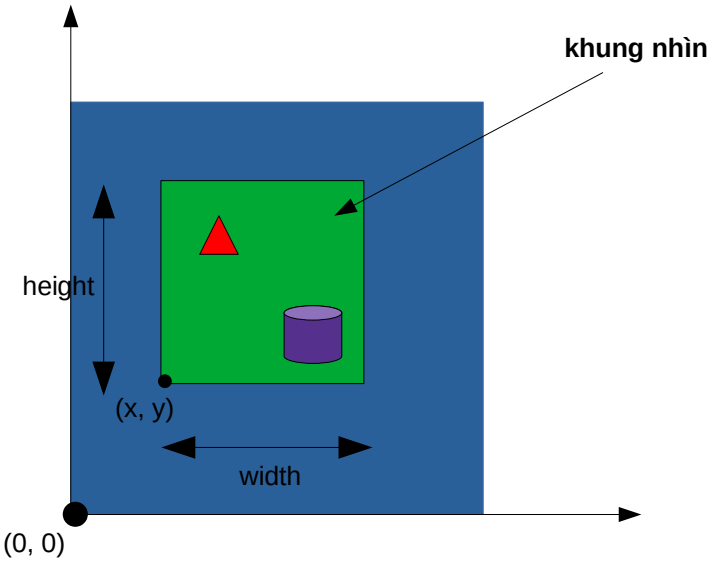


Khung nhìn

- Khung nhìn được định nghĩa dựa trên **trục tọa độ thế giới**
- Để định nghĩa khung nhìn, ta thực hiện câu lệnh sau

gl.viewport(x, y, width, height);

  - **(x, y)**: Điểm tại góc dưới trái của khung nhìn theo trục tọa độ thế giới.
  - **width**: Chiều rộng của khung nhìn.
  - **height**: Chiều cao của khung nhìn.

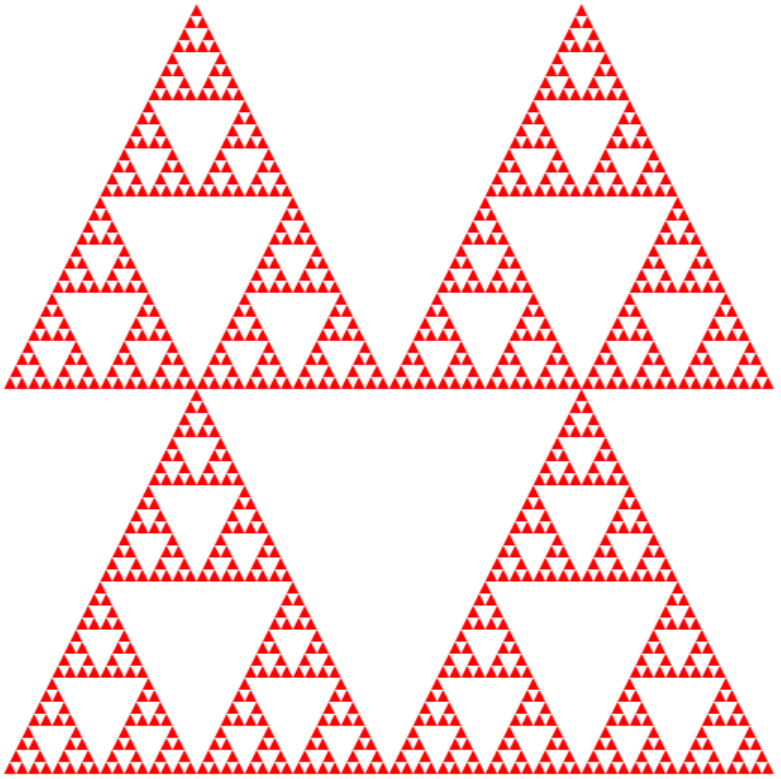


Khung nhìn

- Một số thủ thuật với khung nhìn
  - Tạo ra nhiều khung hình để vẽ nhiều hình vẽ giống nhau.
  - Thay đổi chiều rộng và chiều cao của khung hình để co kéo hình vẽ bên trong.

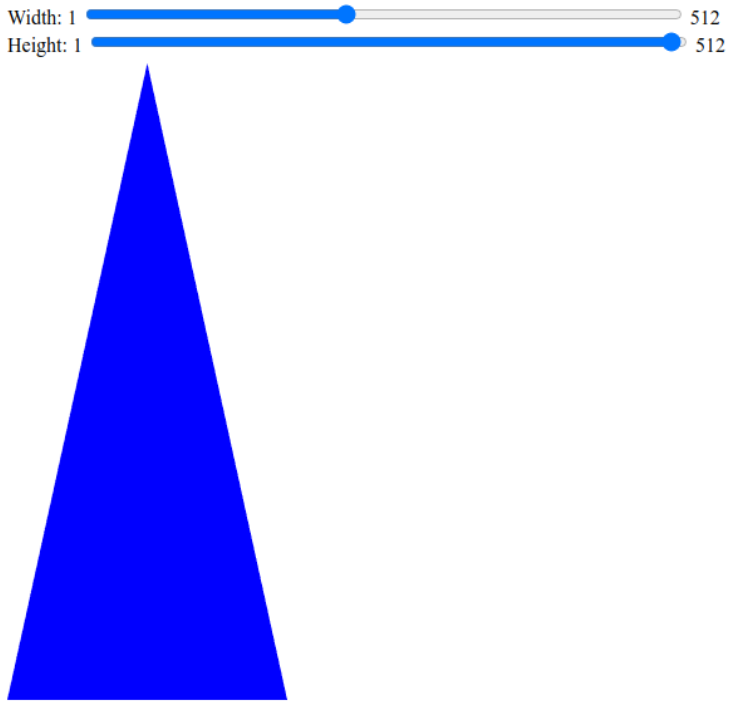
Ví dụ 3

- Sử dụng khung nhìn, vẽ 4 tam giác Sierpinski ở 4 phần của màn hình như sau



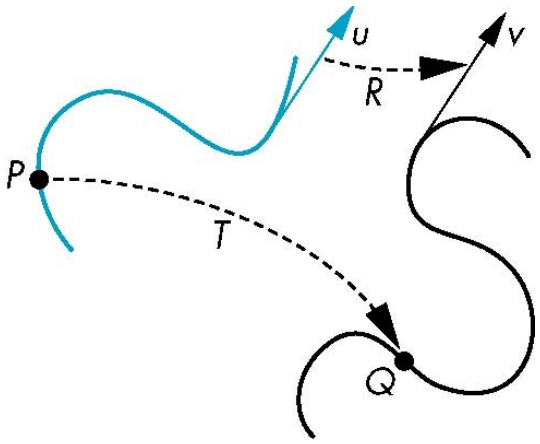
Ví dụ 4

- Viết chương trình thay đổi hình dạng của tam giác bằng cách thay đổi chiều rộng và chiều cao của khung nhìn.



Các phép biến đổi

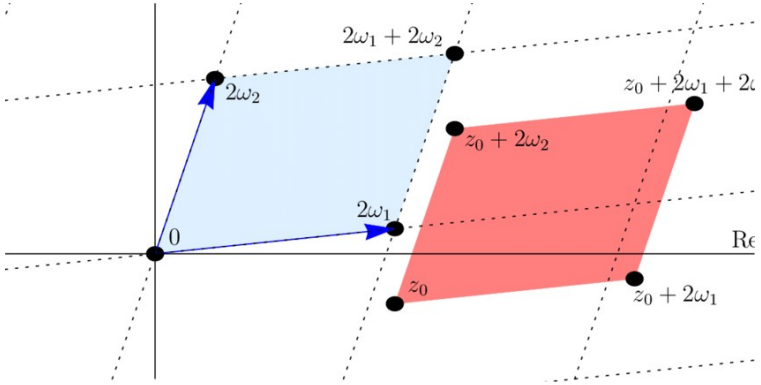
- Các phép biến đổi (transformation) **thực hiện ánh xạ** một điểm đến một điểm khác hoặc một véc-tơ đến một véc-tơ khác.
  - Q = T(P)
  - v = R(u)



  - **T là một ánh xạ** nào đó được dùng để biến đổi điểm P thành điểm Q
    - Ví dụ dịch chuyển một điểm sang vị trí mới.
  - **R là một ánh xạ** nào đó được dùng để biến đổi véc-tơ u thành véc-tơ v
    - Ví dụ thay đổi hướng của một véctơ.

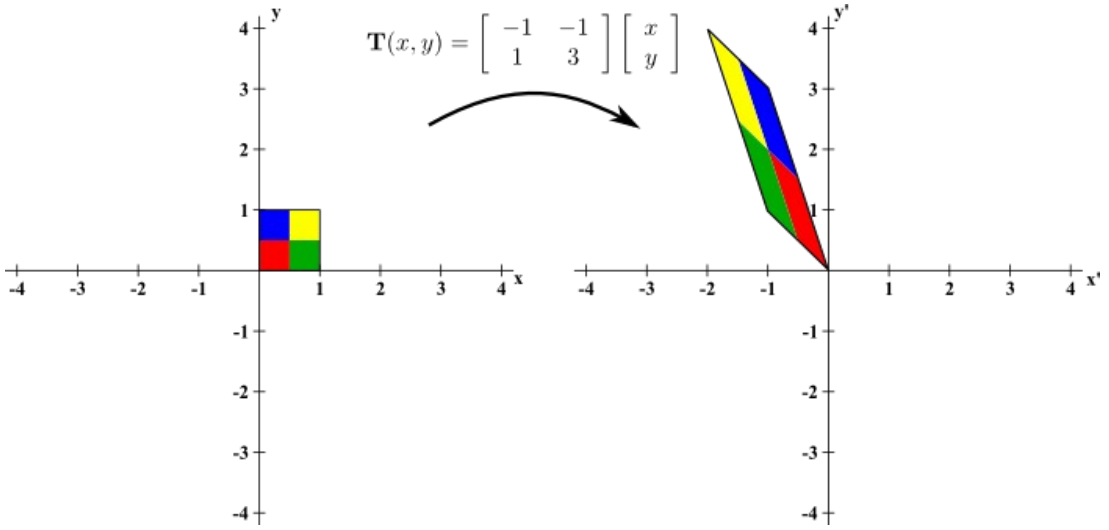
Phép biến đổi Affine

- Một **phép biến đổi Affine** là một phương pháp ánh xạ tuyến tính bảo toàn điểm, đường thẳng và mặt phẳng.
  - Tập hợp các đường thẳng song song sẽ vẫn song song với nhau sau phép biến đổi affine.
    - Ví dụ: Hình bình hành sau phép biến đổi affine thì vẫn là hình bình hành.



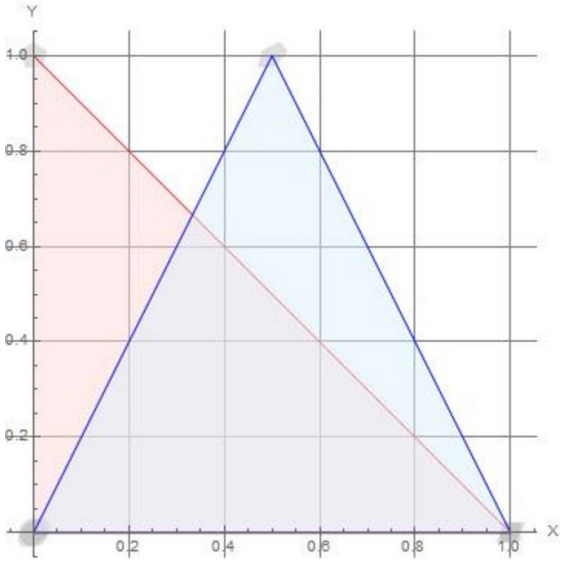
Phép biến đổi Affine

- **Tỷ lệ** trong phép biến đổi affine cũng được bảo toàn
  - Ví dụ: Trung điểm của các đoạn thẳng sau phép biến đổi affine vẫn sẽ là trung điểm của đoạn thẳng mới.



Phép biến đổi Affine

- Tuy nhiên phép biến đổi Affine **không nhất thiết bảo toàn góc và độ dài**.
  - Ví dụ: Một tam giác sau phép biến đổi affine vẫn là một tam giác, nhưng có thể không đồng dạng với tam giác gốc.



Phép biến đổi Affine

- Giả sử trên không gian 2 chiều, một điểm P có tọa độ **(P_x, P_y)**.
- Thông qua một **phép biến đổi T** nào đó, ta cần biến đổi điểm P thành một điểm Q có tọa độ là **(Q_x, Q_y)**
- Trong không gian Affine, tọa độ của Q là **tổ hợp tuyến tính** của tọa độ P (được tính toán dựa trên tọa độ P)

$$Q_x = m_{11}P_x + m_{12}P_y + m_{13}$$

$$Q_y = m_{21}P_x + m_{22}P_y + m_{23}$$

Phép biến đổi Affine

- Dựa vào công thức tổ hợp tuyến tính, ta có 2 cách biểu diễn mối quan hệ giữa P và Q dựa trên ma trận
  - Cách 1:

$$\begin{pmatrix} Q_x \\ Q_y \end{pmatrix} = \begin{pmatrix} m_{11} m_{12} \\ m_{21} m_{22} \end{pmatrix} \begin{pmatrix} P_x \\ P_y \end{pmatrix} + \begin{pmatrix} m_{13} \\ m_{23} \end{pmatrix}$$

  - Cách 2: Sử dụng **ma trận đồng nhất**

$$\begin{pmatrix} Q_x \\ Q_y \\ 1 \end{pmatrix} = \begin{pmatrix} m_{11} m_{12} m_{13} \\ m_{21} m_{22} m_{23} \\ 0 \quad 0 \quad 1 \end{pmatrix} \begin{pmatrix} P_x \\ P_y \\ 1 \end{pmatrix}$$

  - Trong khóa học này ta chọn cách sử dụng **ma trận đồng nhất** và có thể coi ma trận đồng nhất chính là phép biến đổi T (hay ánh xạ T)

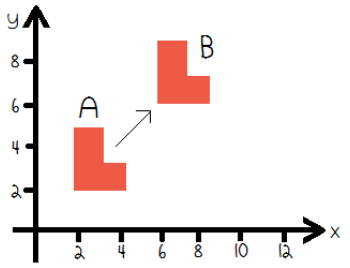
Phép biến đổi hình 2D

- Phép tịnh tiến:
  - Dịch chuyển điểm P sang điểm Q

$$Q_x = P_x + m_{13}$$

$$Q_y = P_y + m_{23}$$

$$\begin{pmatrix} Q_x \\ Q_y \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & m_{13} \\ 0 & 1 & m_{23} \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} P_x \\ P_y \\ 1 \end{pmatrix}$$



  - Ví dụ: Dịch chuyển một điểm theo trục hoành 5 đơn vị và trục tung 2 đơn vị ta sử dụng ma trận đồng nhất bên

$$\begin{pmatrix} 1 & 0 & 5 \\ 0 & 1 & 2 \\ 0 & 0 & 1 \end{pmatrix}$$

Ví dụ 5

- Vẽ một chữ T lên màn hình, sau đó **tịnh tiến chữ T** bằng việc thiết lập các giá trị dịch chuyển cho trục tung và trục hoành.
  - Việc tịnh tiến cả chữ T bản chất là **tịnh tiến các đỉnh** của chữ T sang vị trí mới, sau đó nối các đỉnh mới này lại với nhau.

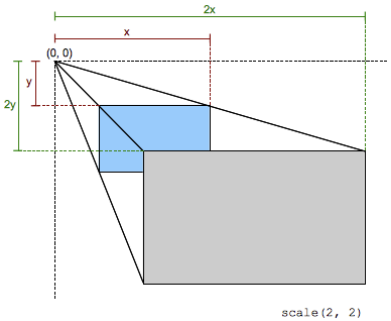
Phép biến đổi hình 2D

- Phép biến đổi tỉ lệ
  - Sử dụng để phóng to, thu nhỏ, kéo dãn, thu hẹp.

$$Q_x = S_x P_x$$

$$Q_y = S_y P_y$$

$$\begin{pmatrix} Q_x \\ Q_y \\ 1 \end{pmatrix} = \begin{pmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} P_x \\ P_y \\ 1 \end{pmatrix}$$



  - Ví dụ: Biến đổi tỷ lệ một hình theo tỉ lệ đối với trục hoành là 1.5 lần và với trục tung là 0.6 lần ta sử dụng ma trận đồng nhất bên

$$\begin{pmatrix} 1.5 & 0 & 0 \\ 0 & 0.6 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Ví dụ 6

- Vẽ một chữ T lên màn hình, sau đó **biến đổi tỷ lệ chữ T** bằng việc thiết lập các giá trị tỷ lệ cho trục tung và trục hoành.
  - Chú ý: Với giá trị nằm trong khoảng (0, 1), hình sẽ bị thu nhỏ.

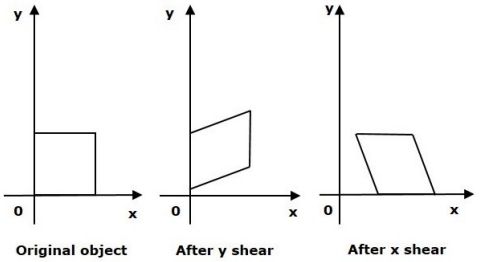
Phép biến đổi hình 2D

- Phép biến dạng
  - Biến đổi hình dạng của đối tượng nhưng vẫn bảo toàn các điểm, đường thẳng và tỷ lệ.

$$Q_x = P_x + R_x P_y$$

$$Q_y = R_y P_x + P_y$$

$$\begin{pmatrix} Q_x \\ Q_y \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & R_x & 0 \\ R_y & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} P_x \\ P_y \\ 1 \end{pmatrix}$$



  - Ví dụ: Biến dạng một hình theo trục hoành là 1.5 và với trục tung là -1.5 ta sử dụng ma trận sau

$$\begin{pmatrix} Q_x \\ Q_y \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & \mathbf{1.5} & 0 \\ \mathbf{-1.5} & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} P_x \\ P_y \\ 1 \end{pmatrix}$$

Ví dụ 7

- Vẽ một chữ T lên màn hình, sau đó **biến dạng chữ T** bằng việc thiết lập các giá trị biến dạng cho trục tung và trục hoành.

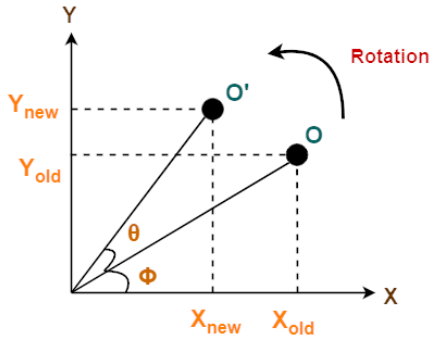
Phép biến đổi hình 2D

- Phép quay quanh gốc tọa độ

$Q_x = \cos(\theta)P_x - \sin(\theta)P_y$

$Q_y = \sin(\theta)P_x + \cos(\theta)P_y$

$$\begin{pmatrix} Q_x \\ Q_y \\ 1 \end{pmatrix} = \begin{pmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} P_x \\ P_y \\ 1 \end{pmatrix}$$



  - Ví dụ: Quay một điểm quanh gốc tọa độ một góc 45 độ ta sử dụng ma trận sau

$$\begin{pmatrix} \sqrt{2}/2 & -\sqrt{2}/2 & 0 \\ \sqrt{2}/2 & \sqrt{2}/2 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Phép biến đổi hình 2D

- Phép quay quanh gốc tọa độ
  - Chứng minh công thức
    - Giả sử ta muốn quay điểm P sang điểm Q
    - Ta có:

      $\cos(\theta + \phi) = Q_x / r$ và $\sin(\theta + \phi) = Q_y / r$

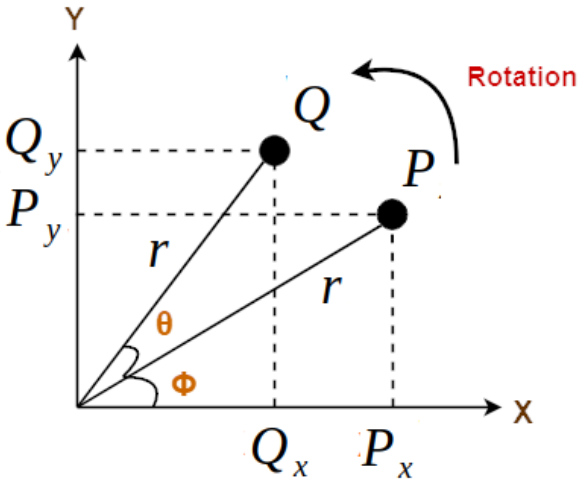
      $\cos(\phi) = P_x / r$ và $\sin(\phi) = P_y / r$
    - Tức là:

      $Q_x = r * \cos(\theta + \phi)$ và $Q_y = r * \sin(\theta + \phi)$ **(1)**

      $P_x = r * \cos(\phi)$ và $P_y = r * \sin(\phi)$ **(2)**
    - Từ (1) ta có :

      $Q_x = r\cos(\theta)\cos(\phi) - r\sin(\theta)\sin(\phi)$

      $Q_y = r\sin(\theta)\cos(\phi) + r\sin(\phi)\cos(\theta)$ **(3)**
    - Thế (2) vào (3) ta có:
      - $Q_x = \cos(\theta) P_x - \sin(\theta) P_y$
      - $Q_y = \sin(\theta) P_x + \cos(\theta) P_y$



Ví dụ 8

- Vẽ một chữ T lên màn hình, sau đó **quay chữ T** quanh gốc tọa độ bằng việc thiết lập các góc quay.
  - Chú ý:
    - Thiết lập góc quay từ 0 độ cho đến 360 độ.
    - Sử dụng các hàm **Math.cos**, **Math.sin** và **Math.PI** có sẵn trong thư viện của Javascript.
      - Đầu vào của các hàm này là các tham số dạng Radian

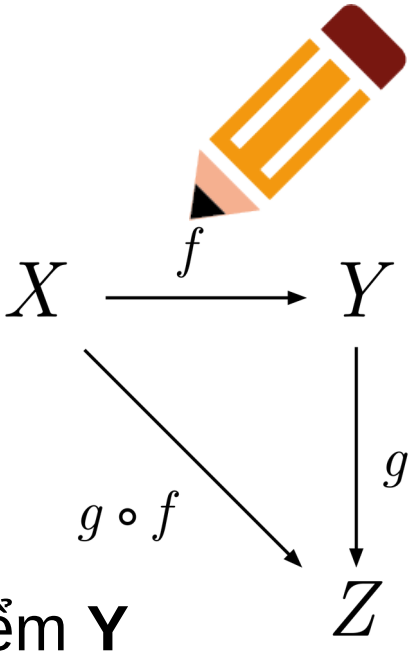
        1 radian = 180 độ / π

        Do đó: **x** radian = **y** độ * π / 180

Phép biến đổi hình 2D

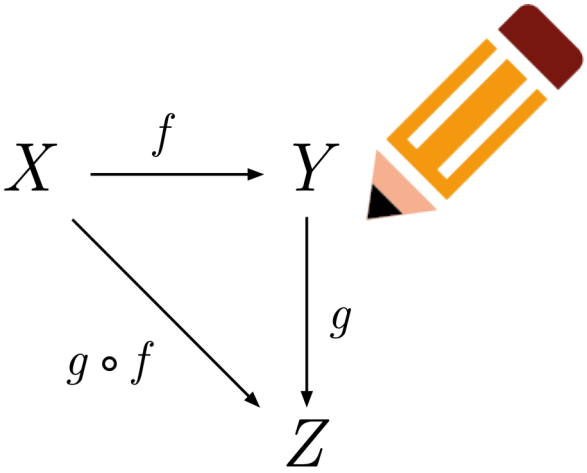
- Phép biến đổi tổ hợp
  - Trong thực tế, ta thường phải thực hiện **các phép biến đổi phức tạp**, tuy nhiên mọi phép biến đổi phức tạp đều có thể được tạo thành từ các phép biến đổi cơ sở ở trên, do đó ta gọi đây là **phép biến đổi tổ hợp**.
    - Ví dụ:
      - Vừa di chuyển hình, vừa xoay hình.
      - Vừa xoay hình, vừa biến đổi tỉ lệ hình
      - Quay một hình quanh một điểm không phải là gốc tọa độ.

Phép biến đổi hình 2D



- Phép biến đổi tổ hợp
  - Giả sử ta có
    - Điểm **X** qua phép biến đổi **f** ta nhận được điểm **Y**
      - **$Y = f(X) = f_{matrix} * X$**
    - Điểm **Y** qua phép biến đổi **g** ta nhận được điểm **Z**
      - **$Z = g(Y) = g_{matrix} * Y$**
  - Câu hỏi:
    - **Có tồn tại phép biến đổi nào để biến đổi ngay lập tức điểm X thành điểm Z được không ?**
  - Trả lời:
    - **Có tồn tại**. Đó là hợp của hai phép biến đổi g và f

Phép biến đổi hình 2D



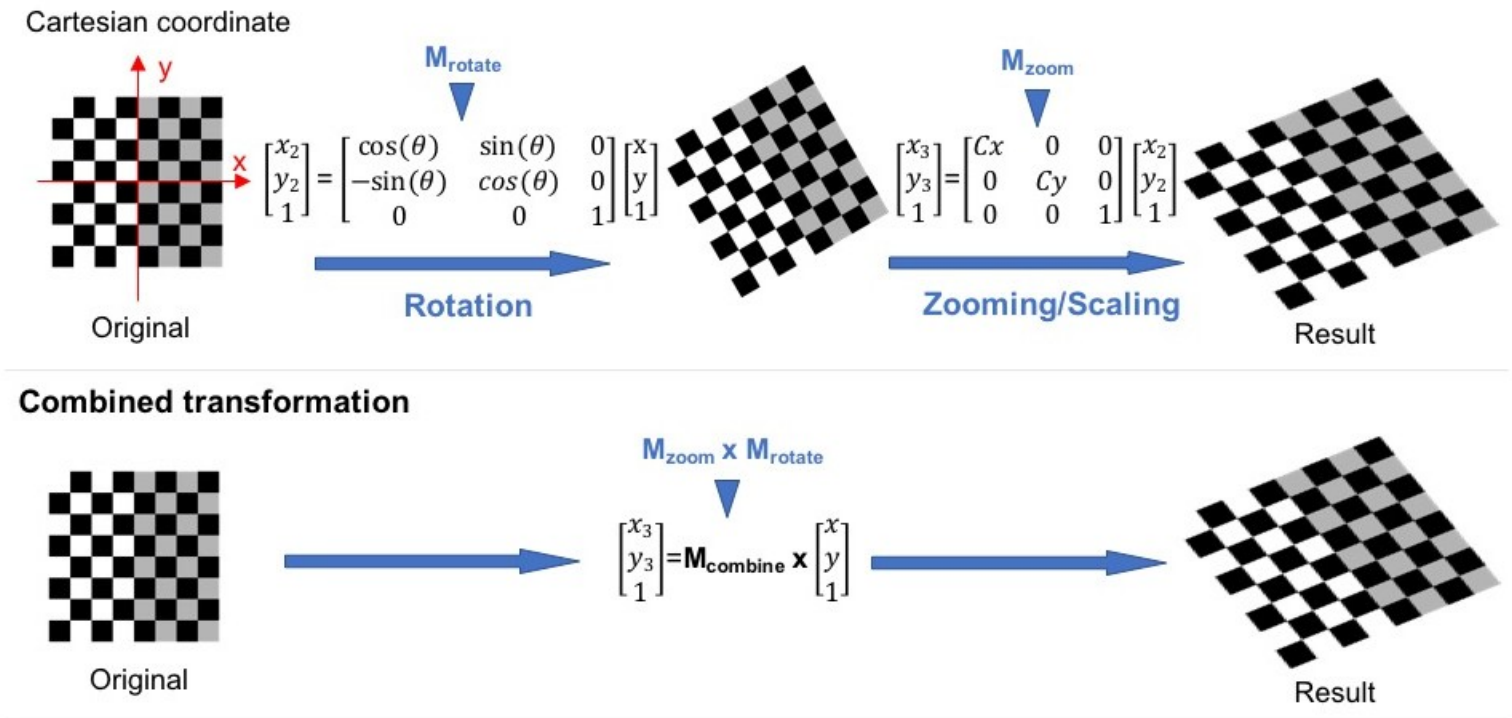
- Phép biến đổi tổ hợp
  - Ta cần phải **tìm ma trận biến đổi T_{matrix}** để ánh xạ điểm X thành điểm Z.
  - Cách tìm:
    - Ánh xạ f là ma trận biến đổi điểm X thành điểm Y.
    - Ánh xạ g là ma trận biến đổi điểm Y thành điểm Z.
    - Ta có: $Z = g_{matrix} * Y = g_{matrix} * (f_{matrix} * X) = (g_{matrix} * f_{matrix}) * X$
    - Vậy $T_{matrix} = g_{matrix} * f_{matrix}$
  - Kết luận:
    - Ta thấy rằng để tính ma trận biến đổi T_{matrix} ta chỉ cần lấy ma trận ánh xạ g nhân với ma trận ánh xạ f (theo đúng thứ tự g đứng trước, f đứng sau)

Phép biến đổi hình 2D

- Phép biến đổi tổ hợp
  - Phương pháp tìm phép biến đổi tổ hợp
    - B1: Phân tích cách thức biến đổi ra thành các phép biến đổi cơ bản.
      - Đã có ma trận cho các phép biến đổi cơ bản ở phần trên.
    - B2: Lấy ma trận của các phép biến đổi cơ bản nhân với nhau.
      - Lưu ý về trình tự: **Phép biến đổi cơ bản nào thực hiện trước thì việc nhân ma trận sẽ được thực hiện sau**.

Phép biến đổi hình 2D

- Phép biến đổi tổ hợp
  - Ví dụ 1:

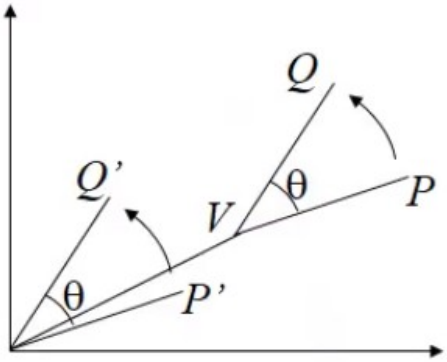


Phép biến đổi hình 2D

- Phép biến đổi tổ hợp
  - Ví dụ 2: Xây dựng ma trận biến đổi để xoay một điểm $P(P_x, P_y)$ quanh một điểm $V(V_x, V_y)$ không phải gốc tọa độ.
    - **Tìm ma trận tổ hợp T**:
      - B1: Sau khi phân tích, cách thức biến đổi này có 3 phép biến đổi cơ bản.
        - Thực hiện tịnh tiến điểm P theo véc-tơ $v = (-V_x, -V_y)$ thành điểm P' (ký hiệu ma trận biến đổi là $\mathbf{f}_{matrix}$)

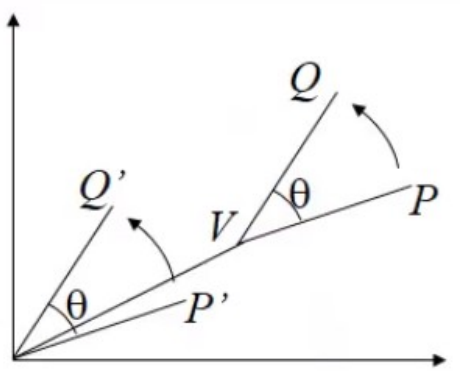
          Quay điểm P' quanh gốc tọa độ thành điểm Q' (ký hiệu ma trận biến đổi là $\mathbf{g}_{matrix}$)

          Thực hiện tịnh tiến điểm Q' theo véc-tơ $v' = (V_x, V_y)$ ta được điểm Q (đây chính là điểm ta muốn điểm P xoay đến) (ký hiệu ma trận biến đổi này là $\mathbf{h}_{matrix}$)



Phép biến đổi hình 2D

- Phép biến đổi tổ hợp
  - Ví dụ 2:
    - **Tìm ma trận tổ hợp T**:
      - B2: $T_{matrix} = h_{matrix} \times g_{matrix} \times f_{matrix}$



$$\begin{pmatrix} 1 & 0 & V_x \\ 0 & 1 & V_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & -V_x \\ 0 & 1 & -V_y \\ 0 & 0 & 1 \end{pmatrix}$$

$$= \begin{pmatrix} \cos(\theta) & -\sin(\theta) & (1-\cos(\theta)) * V_x + \sin(\theta) * V_y \\ \sin(\theta) & \cos(\theta) & -\sin(\theta) * V_x + (1-\cos(\theta)) * V_y \\ 0 & 0 & 1 \end{pmatrix}$$

Ví dụ 9

- Vẽ một chữ T lên màn hình, sau đó **quay chữ T** quanh một điểm cho trước bằng việc thiết lập các góc quay.
  - Chú ý:
    - Thiết lập góc quay từ 0 độ cho đến 360 độ.

Hết Tuần 3

Cảm ơn các bạn đã chú ý lắng nghe !!!