Lucrarea de laborator Nr. 2

Tema: Transformări geometrice a figurilor grafice 2D și 3D

Scopul lucrării: Obţinerea cunoştinţelor practice în realizarea transformărilor geometrice de scalare, deplasare, și rotire a figurilor 2D și 3D. Aplicarea matricilor de transformare a figurilor geometrice, utilizînd mediul on-line https://editor.p5js.org/.

Sarcina lucrării: De creat o scenă grafică dinamică 2D și 3D, utilizând metodele de transformare din mediul JS - *applyMatrix(), resetMatrix(), rotate(), rotateX(), rotateY(), rotateZ(), scale(), shearX(), shearY(), translate()*.

Mersul lucrării:

1. De a elaborat un program pentru sintezarea și afișarea unei scene grafice 2D dinamice utilizînd transformările grafice elementare, utilizînd metodele respective din mediul on-line https://editor.p5js.org/;

2. De a elaborat un program pentru sintezarea și afișarea unei scene grafice 3D dinamice utilizînd primitivele și transformările grafice elementare în mediul on-line https://editor.p5js.org/;

3. De elaborat raportul la lucraea de laborator, care trebuuie să conțină următoarele: foaie de titlu (Numele prenumele grupa și tema lucrării); scopul lucrării; sarcina propusă spre elaborare; codul programului; rezultatele programului; concluzii.

4. Raportul se prezintă în format electronic pe e-mail: mariana.osovschi@calc.utm.md, pînă la începutul sesiunii de examinare.

Întrebări de verificare:

1. Translaţia în plan
2. Rotaţia în jurul originii
3. Rotaţia imaginilor raster
4. Transformarea de scalare neuniformă
5. Imaginea în oglinda a unui obiect
6. Secvenţa de transformări
7. Transformări geometrice 2D și 3D elementare
8. Scalarea faţă de origine

9. Rotaţie în jurul unei axe a sistemului de coordonate
10. Afişarea unui obiect 2D, 3D pe ecran

Considerații teoretice

Функция rotate(): поворот фигуры на величину, указанную параметром угла. Эта функция учитывает angleMode, поэтому углы можно вводить в RADIANS или DEGREES.

Объекты всегда вращаются вокруг своей относительной позиции к началу координат, а положительные числа вращают объекты по часовой стрелке. Преобразования применяются ко всему, что происходит после, и последующие вызовы функции накапливают эффект. Например, вызов rotate (HALF_PI), а затем rotate (HALF_PI) аналогичен rotate (PI). Все преобразования сбрасываются, когда draw () начинается снова.

Технически rotate () умножает текущую матрицу преобразования на матрицу вращения. Эту функцию можно дополнительно контролировать с помощью push () и pop ().

Синтаксис

```
rotate(angle, [axis])
```

Параметры

angle: угол поворота, указанный в радианах или градусах, в зависимости от текущего angleMode

axis: (в 3d) задаём ось для вращения вокруг (необязательно)

Функции rotateX(), rotateY(), rotateZ(): вращение вокруг оси X, Y и Z соответственно.

Синтаксис

```
rotateX(angle)

rotateY(angle)

rotateZ(angle)
```

Параметры

angle: угол поворота, указанный в радианах или градусах, в зависимости от текущего angleMode.

Функция scale(): увеличивает или уменьшает размер фигуры, расширяя и сужая вершины. Объекты всегда масштабируются от их относительного происхождения до системы координат. Значения шкалы указываются в десятичных процентах. Например, масштаб вызова функции (2.0) увеличивает размер фигуры на 200%.

Преобразования применяются ко всему, что происходит после и последующие вызовы функции умножают эффект. Например, вызов масштаба (2.0), а затем масштаба (1.5) такой же, как масштаб (3.0). Если scale () вызывается в draw (), преобразование сбрасывается, когда цикл начинается снова.

Использование этой функции с параметром z доступно только в режиме WEBGL. Эту функцию можно дополнительно контролировать с помощью push () и pop ().

Синтаксис

```
scale(s, [y], [z])

scale(scales)
```

Параметры

s: процент для масштабирования объекта или процент для масштабирования объекта по оси X, если задано несколько аргументов

y: процент для масштабирования объекта по оси y (необязательно)

z: проценты для масштабирования объекта по оси z (только webgl) (необязательно)

scale: проценты по оси для масштабирования объекта

Функция shearX(): обрезает форму вокруг оси X на величину, указанную параметром угла. Углы должны быть указаны в текущем angleMode. Объекты всегда срезаются вокруг их относительного положения относительно исходного положения, а положительные числа срезают объекты по часовой стрелке.

Преобразования применяются ко всему, что происходит после, и последующие вызовы функции накапливают эффект. Например, вызов shearX (PI / 2) и затем shearX (PI / 2) аналогичен shearX (PI). Если в draw () вызывается shearX (), преобразование сбрасывается, когда цикл начинается снова.

Технически shearX() умножает текущую матрицу преобразования на матрицу вращения. Эта функция может далее управляться функциями push () и pop ().

Синтаксис

```
shearX(angle)
```

Параметры

angle: угол сдвига, указанный в радианах или градусах, в зависимости от текущего angleMode.

Функция shearY(): обрезает форму вокруг оси Y на величину, указанную параметром угла. Углы должны быть указаны в текущем angleMode. Объекты всегда срезаются вокруг их относительного положения относительно исходного положения, а положительные числа срезают объекты по часовой стрелке.

Преобразования применяются ко всему, что происходит после, и последующие вызовы функции накапливают эффект. Например, вызов shearY (PI / 2) и затем shearY(PI / 2) аналогичен shearY (PI). Если в draw() вызывается shearY (), преобразование сбрасывается, когда цикл начинается снова.

Технически shearY () умножает текущую матрицу преобразования на матрицу вращения. Эта функция может далее управляться функциями push() и pop().

Синтаксис

```
shearY(angle)
```

Параметры

angle: угол сдвига, указанный в радианах или градусах, в зависимости от текущего angleMode.

Функция translate(): определяет смещение объектов в окне отображения. Параметр x указывает перевод влево/вправо, параметр y - перевод вверх/вниз.

Преобразования являются кумулятивными и применяются ко всему, что происходит после, и последующие вызовы функции накапливают эффект. Например, вызов translate (50, 0) и затем translate (20, 0) аналогичен translate

(70, 0). Если translate () вызывается в draw (), преобразование сбрасывается, когда цикл начинается снова. Эту функцию можно дополнительно контролировать с помощью push () и pop ().

Синтаксис

```
translate(x, y, [z])
```

```
translate(vector)
```

Параметры

x: левый / правый перевод

y: перевод вверх / вниз

z: прямой / обратный перевод (только webgl) (необязательно)

vector: задаёт вектор сдвига

Функция applyMatrix(): Умножает текущую матрицу на ту, которая указана в параметрах. Это мощная операция, которая может выполнять эквивалент сдвига, масштабирования, сдвига и поворота одновременно. Вы можете узнать больше о матрицах преобразования в конспекте по графике.

Именование аргументов здесь следует за наименованием спецификации WHATWG и соответствует матрице преобразования вида:

Матрица преобразования, используемая при вызове applyMatrix

$$\begin{bmatrix} a & c & e \\ b & d & f \\ 0 & 0 & 1 \end{bmatrix}$$

Синтаксис

```
applyMatrix(a, b, c, d, e, f)
```

Параметры

a: числа, которые определяют умножаемую матрицу 2x3

b: числа, которые определяют умножаемую матрицу 2x3

c: числа, которые определяют умножаемую матрицу 2x3

d: числа, которые определяют умножаемую матрицу 2x3

e: числа, которые определяют умножаемую матрицу 2x3

f: числа, которые определяют умножаемую матрицу 2x3

Функция resetMatrix(): Заменяет текущую матрицу на единичную матрицу.

Синтаксис

```
resetMatrix()
```