Розробка програмного забезпечення із доповненою реальністю з використанням JavaScript

Мета – навчитися створювати програми із доповненою реальністю, орієнтовані на традиційні та мобільні веб-браузери, за допомогою бібліотек Three.js та AR.js.

Пререквізити – успішне опанування курсу «Web-програмування».

Обладнання – стаціонарний або мобільний комп'ютерний засіб, обладнаний зовнішньою або вбудованою камерою.

Програмне забезпечення – веб-браузер із підтримкою WebGL, текстовий редактор.

Анотація. Ідея доповненої реальності (Augmented Reality – AR) полягає у комбінуванні комп'ютерно-генерованих моделей з об'єктами реального світу. AR.js надає набір засобів для ефективної розробки AR-програм за допомогою веб-технологій, таких як JavaScript і HTML. Вступ містить основ AR та короткий огляд усіх бібліотек та засобів розробки. Робота розпочинається з налаштування базової структури AR-програми з використанням бібліотеки AR.js. Далі виконується налаштування камери для локалізації маркерів AR. Після цього виконується накладання 3D-контенту поверх маркерів за допомогою бібліотеки Three.js з використанням реалістичного освітлення та тіней до 3D-об'єктів з різними елементами керування, що надаються Three.js для навігації по 3D-сцені. Наприкінці пропонується кілька проектів AR-програм для закріплення набутих навичок створення AR-програм за допомогою A-Frame, Three.js, AR.js та awe.js.

1 ТЕХНОЛОГІЇ ДОПОВНЕНОЇ РЕАЛЬНОСТІ

1.1. Вступ

Програмувати доповнену реальність – інноваційно (модно, цікаво, корисно та ін.) останні 60 років, і вибір JavaScript в якості мови програмування – виключно данина часу. Але вибір мови визначає й вибір засобів розробки, найбільш доцільними з яких на сьогодні є:

а) A-Frame та AR.js – ці API, фактично, унікальні засоби швидкого прототипування, і значна частина програми з їх використанням – це HTML-подібний код, який використовує JavaScript на сервері. A-Frame використовується для створення сцен, об'єктів, анімації та інших 3D-елементів у веб-браузері. AR.js надає можливість відслідковувати маркер і надає можливість сцені, сконструйованій за допомогою A-Frame, відображатися прямо на маркері;

б) Three.js – своєрідний кістяк, який використовуються багато інших бібліотек мовою JavaScript. Three.js використовує WebGLRenderer, що надає можливість створення якісних 3D-сцени безпосередньо у браузері. На відміну від A-Frame, Three.js використовується в основному для створення веб-програм під управлінням Google Cardboard та вимагає явного використання JavaScript;

в) Awe.js: AR API, що використовує дуже схожу з Three.js мову та виконує таку саму роль по відношенню до Three.js, як AR.js стосовно A-Frame.

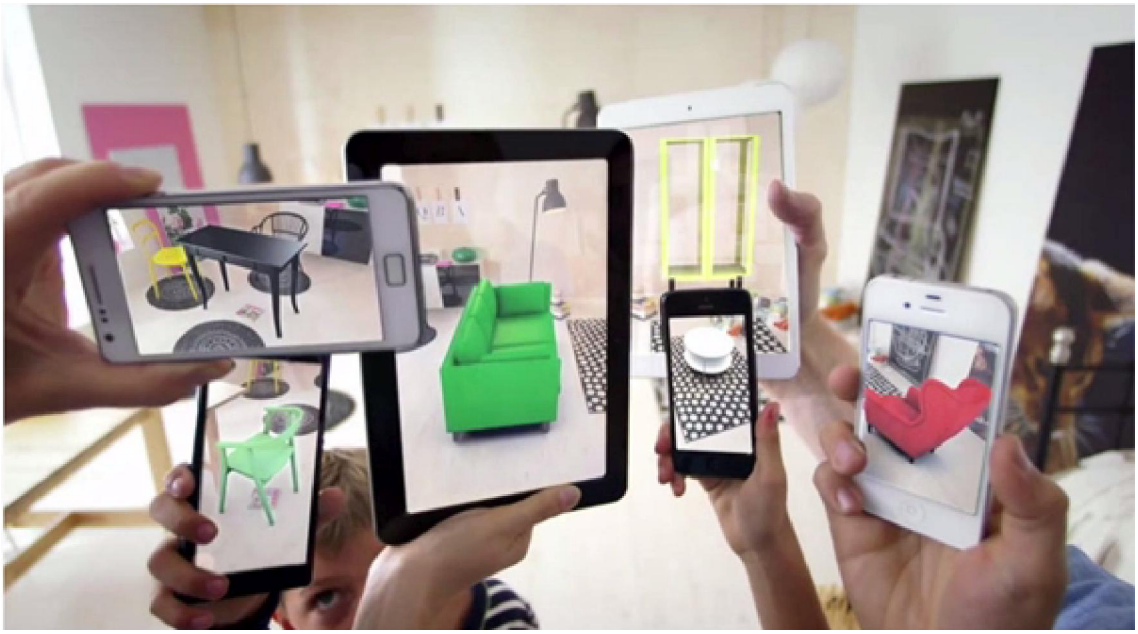
Програмні засоби із доповненою реальністю, розроблені із використанням JavaScript та WebGL, можуть бути розміщені в Інтернет на одному із хмарних сервісів, таких як Heroku.

1.2. Як працює доповнена реальність

Для початку роботи необхідно мати лише AR-сумісний браузер, такий як Firefox або Chrome, та текстовий редактор (типу mcedit або Sublime). Базові знання HTML, CSS та JavaScript є обов'язковими, а досвід роботи з веб-API та

GitHub стануть у пригоді.

Надалі під AR будемо розуміти здатність пристрою, зокрема мобільного пристрою або веб-браузера, відстежувати зображення або відображати 3D-об’єкт поверх цього зображення. Головна ідея AR полягає в тому, щоб відобразити комп’ютерну модель у реальному часі та реальному просторі з метою взаємодії між користувачем у реальному просторі та 3D-моделі у віртуальному.



AR може бути як маркерним, так й безмаркерним. У маркерній AR пристрій відстежує 2D-маркер: коли він знаходиться, на ньому фактично відображається 3D-об’єкт. У безмаркерному варіанті пристрій буде шукати плоску поверхню (стіл, підлогу тощо), і розташовуватимемо 3D-об’єкт на ній.

Використовуючи камеру пристрою, AR надає можливості відображення комп’ютерно згенерованих об’єктів в ігрових, маркетингових та інших програмах – наприклад, для розстановки меблів у вітальні або примірки одягу перед їх покупкою. Це дійсно велика можливість для бізнесу – показати, як виглядає продукція, перш ніж будь-який споживач дійсно її купує.

Для AR розробляються спеціальні пристрої, як правило, у вигляді шоломів та гарнітур, що надають можливість занурення користувача у модельне середовище.

1.3. Порівняння доповненої та віртуальної реальності

AR доповнює реальний світ 3D-моделями, якими можна керувати за допомогою мобільного пристрою в будь-якому місці. Віртуальна реальність (Virtual Reality – VR) занурює користувача у модельний світ, для чого, як правило, необхідні наголовні дисплеї (Head Mounted Devices – HMD).

Інтерактивність у програмах для AR і VR забезпечується дуже схоже. Так, наприклад, VR фактично використовує контролери, а у деяких випадках

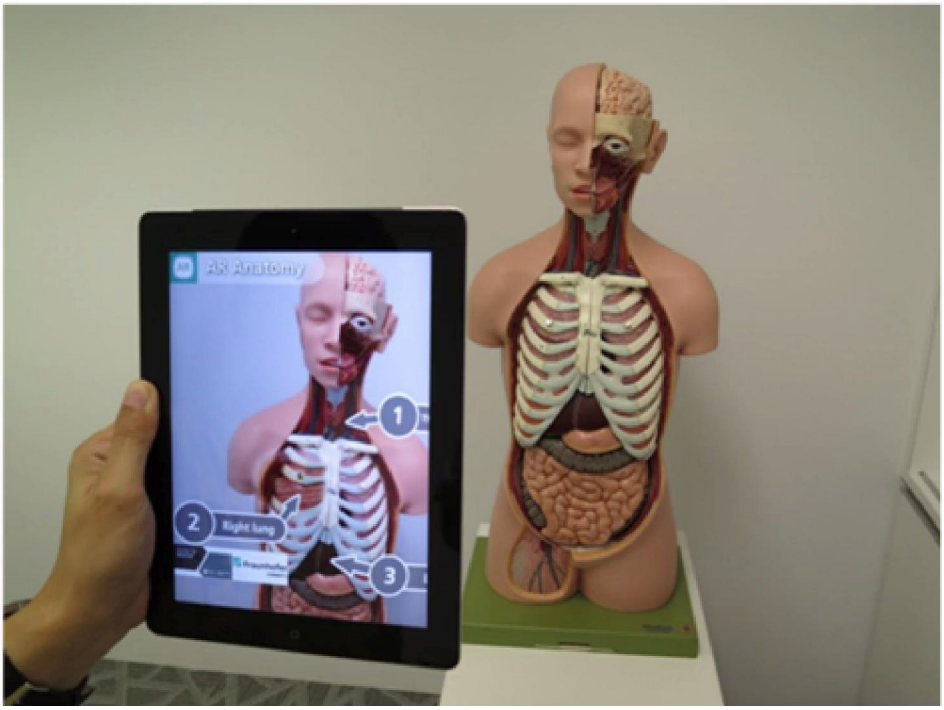
й відстеження рук, що надає змогу користувачеві взаємодіяти з 3D-об’єктами всередині сцени, у якій вони знаходяться.

До головних небезпек використання HMD для роботи у VR відносяться:

– напруження очей;

– запаморочення і головні болі після використання HMD.

На відміну від VR, AR не має таких значних ризиків для здоров’я. Тим не менш, викликає занепокоєння можливість користувачів залишатися зосередженими на тому, що вони роблять, під час використання AR – зокрема, з причин безпеки.



1.4. Пристрої доповненої реальності

Найбільш поширений тип пристроїв, готових для AR – смартфони та планшети з операційними системами iOS (версія 11 та вище під управлінням iPhone та iPad) та Android (версія 7 та вище).

Для веб-браузерів AR доступна, якщо у них реалізована підтримка WebRTC та WebGL – насамперед Google Chrome та Mozilla Firefox.

HoloLens є HMD-подібною гарнітурою для AR, що знаходиться у активній розробці. Цю гарнітуру часто порівнюють з AR-окулярами, такими як CastAR, Meta, Laster SeeThru та K-Glass.

2 СТВОРЕННЯ ПРОСТИХ AR-ПРОГРАМ ЗА ДОПОМОГОЮ A-FRAME ТА AR.JS

2.1. Налаштування проекту та створення сцени

A-Frame дуже схожий на HTML – усі команди описуються тегами, які подібні до тегів HTML, але, на відміну від останніх, інтерпретуються не у веб-браузері на боці клієнта, а є способом доступу до JavaScript, що виконується на боці сервера. Разом із AR.js він є потужним API для AR, що приховує деталі реалізації мовою JavaScript.

Для початку роботи з A-Frame необхідно перейти на сайт A-Frame за посиланням https://aframe.io/docs/1.0.0/introduction/installation.html та завантажити з нього файл збірки JavaScript (JS Build, Production Version – https://aframe.io/releases/1.0.3/aframe.min.js).

Завантажений файл збірки необхідно зберегти у окремому каталозі (наприклад, aframe) всередині робочого каталогу, після чого створити у останньому індексний файл HTML ARindex.html:

```
code
    ARindex.html

    aframe
        aframe.min.js
```

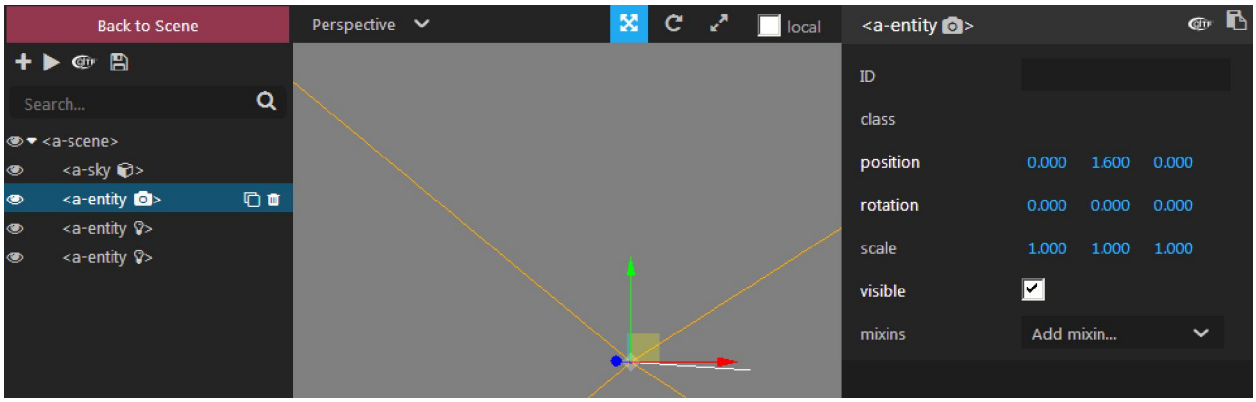
із наступним вмістом

```
<!DOCTYPE html>
```

```
<html>
  <script src="aframe/aframe.min.js"></script>
  <a-scene>
    <a-sky color="grey"></a-sky>
  </a-scene>
</html>
```

Файл містить команду `script` із посиланням на завантажений файл – вона завжди використовується для початку роботи з бібліотекою JavaScript. Саме у ньому інтерпретується тег `a-scene`, в якому знаходитиметься більша частина коду A-Frame. Тег `a-sky` створюватиме фоновий колір (його також можна застосувати для розміщення 360° зображення на сцені). Після відкриття файлу ARindex.html у веб-браузері отримаємо вікно із сірим фоном.

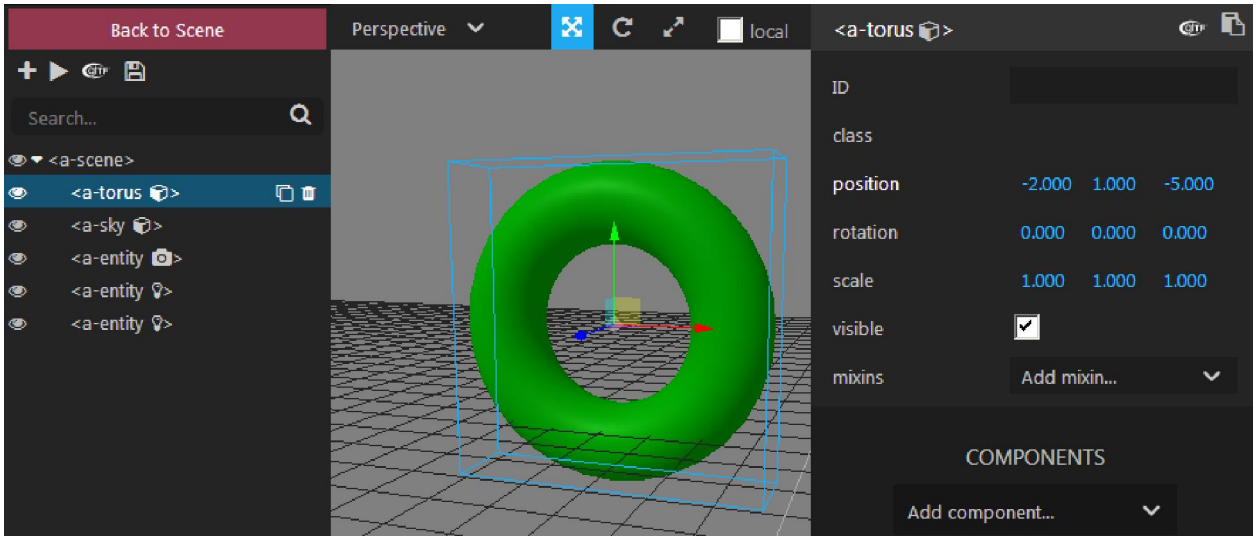
Про те, що A-Frame API дійсно працює, свідчить режим VR (Virtual Reality) у нижньому правому куті. Суттєво більше відомостей можна отримати, увімкнувши режим візуального 3D-інспектора (Ctrl-Alt-I).



Додавши до індексного файлу команду

```
<a-torus position="-2 1 -5" color="green" radius="1.2"></a-torus>
```

отримаємо зображення зеленого тору на сірому тлі.



У новостворених об’єктів є певні атрибути – ознайомитись із ними можна у документації до A-Frame. Поточна версія підтримує наступні примітиви:

`a-box` – прямокутний паралелепіпед;

`a-camera` – камера (визначає, що бачить користувач);

`a-circle` – круг;

`a-collada-model` – 3D-модель у форматі COLLADA (.dae);

`a-cone` – конус;

`a-cursor` – опрацювання подій від миши;

`a-curvedimage` – панорамне зображення;

`a-cylinder` – циліндричні поверхні;

`a-dodecahedron` – дванадцятигранник;

`a-gltf-model` – 3D-модель у форматі glTF (.gltf);

`a-icosahedron` – двадцятигранник;

`a-image` – пласке зображення;

`a-light` – джерела світла;

`a-link` – гіперпосилання;

`a-obj-model` – 3D-модель у форматі Wavefront (.obj/.mtl);

`a-octahedron` – восьмигранник;

`a-plane` – площина;

`a-ring` – пласке кільце або диск;

`a-sky` – додає фоновий колір або 360° зображення;

`a-sound` – джерело звуку;

`a-sphere` – сфера або багатогранник;

`a-tetrahedron` – трикутна піраміда;

`a-text` – плаский текст;

`a-torus-knot` – кренделеподібна фігура;

`a-torus` – тор;

`a-triangle` – трикутна поверхня;

`a-video` – відео як текстура на площині;

`a-videosphere` – 360° фонове відео.

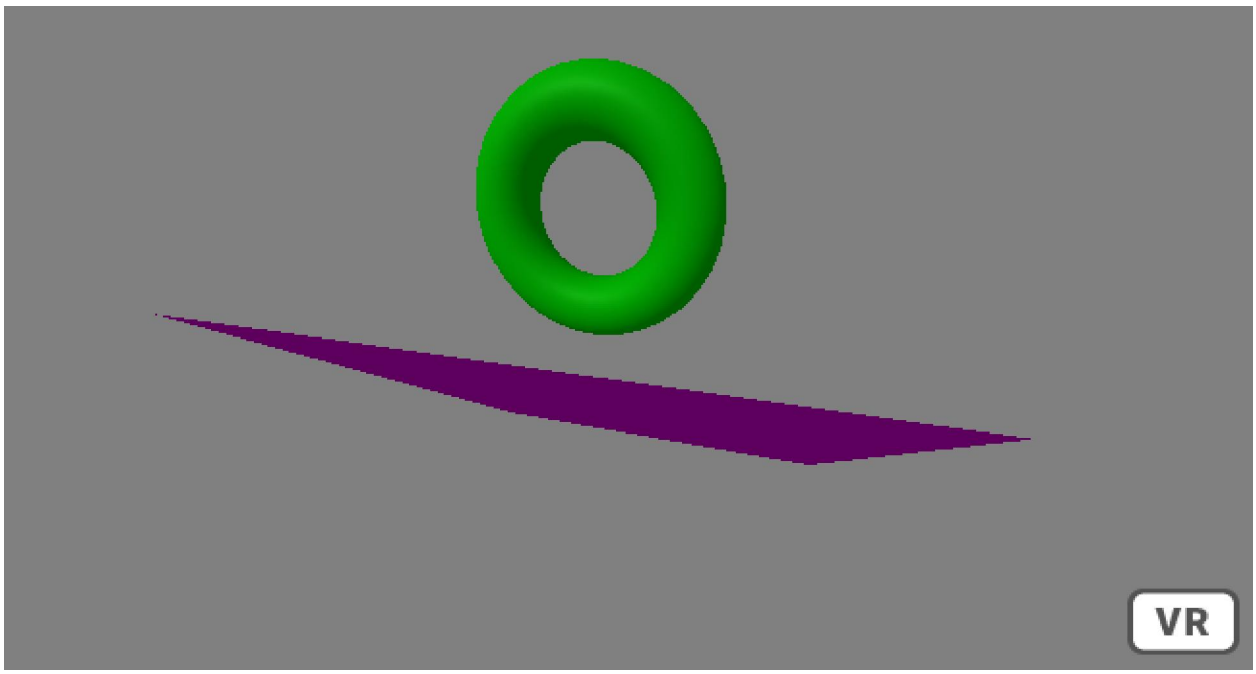
2.2. Об’єктні сітки та атрибути сітки

Додамо до попередньої сцени, що містить зелений тор та сірий фон, ще кілька примітивів. Спочатку вставимо між тором та фоном площину:

```
<a-plane width="7" height="7" rotation="50 0 0"
         position="3 -2 -3" color="purple"></a-plane>
```

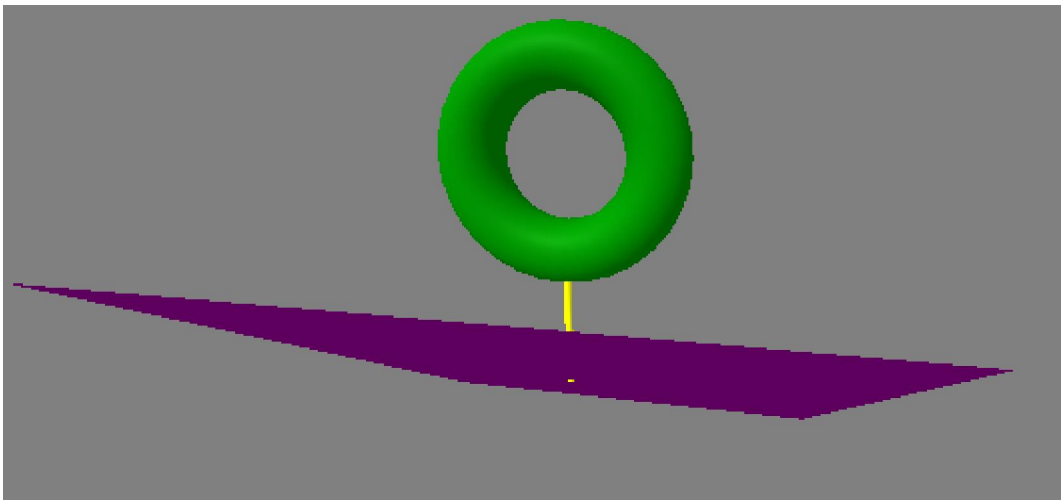
Для перегляду параметрів `a-plane` слід скористатись документацією за посиланням https://aframe.io/docs/1.0.0/primitives/a-plane.html – тут можна знайти різні атрибути площини, які можна застосувати до неї в тегу A-Frame. Параметри `width` та `height` відповідають за ширину та висоту прямокутника, `rotation` вказує на необхідність її повороту на 50° відносно вісі *x* та на 0 – відносно *y* та *z*, `position` – координати початку площини за відповідними осями, а `color` – обраний колір.

Площину, налаштовану у такий спосіб, на сцені можна побачити лише в режимі інспектора, тому що вона розміщена дуже далеко на задньому плані. Якщо змінити координати початку площини на "-2 -2 -5", її можна побачити.



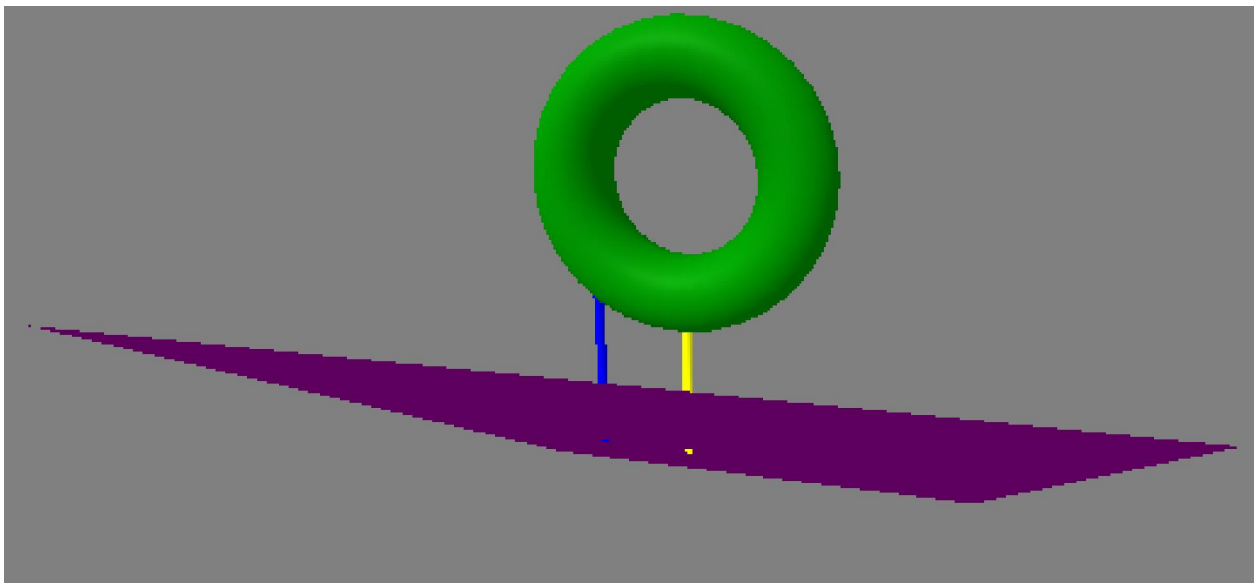
Отже, якщо в певній позиції об’єкт фактично не відображається, це може бути проблемою з позиціонуванням самого об’єкту всередині коду. Тепер, коли у нас є площина, додамо ще кілька об’єктів. Створимо циліндричний об’єкт у формі льодяника жовтого кольору, розташованого під самим тором

```
<a-cylinder color="yellow" height="2" radius="0.05"
            position="-2 -1 -5"></a-cylinder>
```



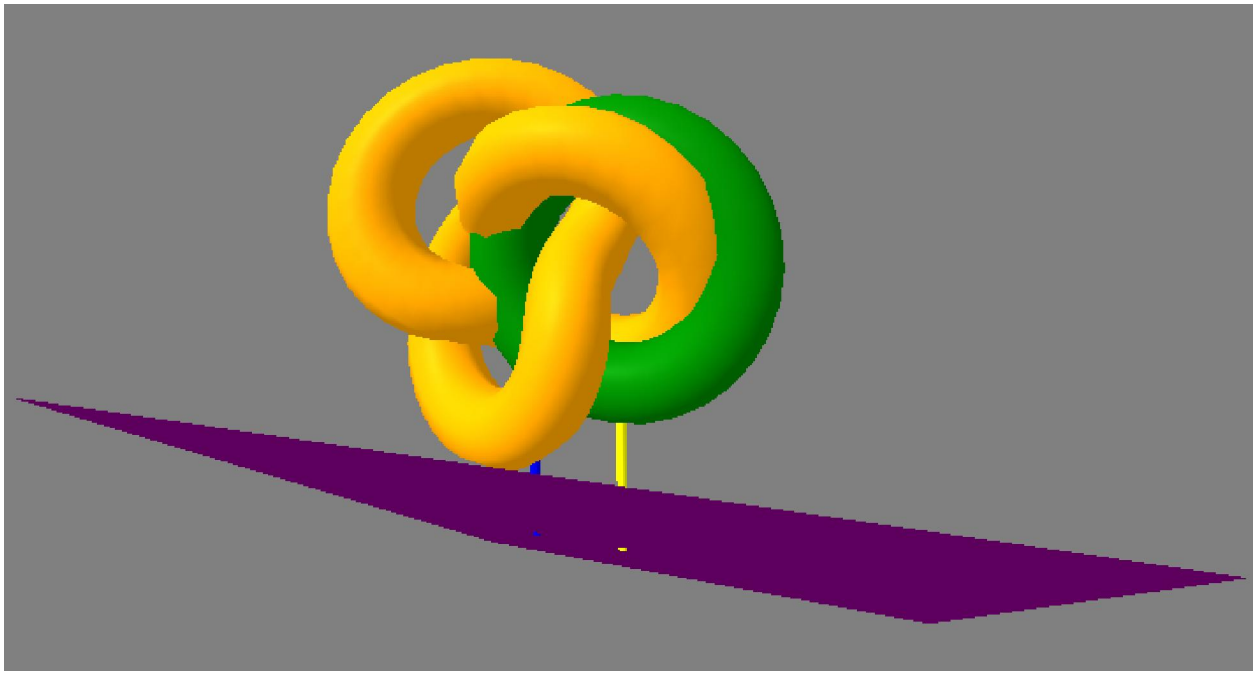
Ще один циліндр:

```
<a-cylinder color="blue" height="2" radius="0.05"
            position="-3 -1 -5"></a-cylinder>
```



І, нарешті, кренделеподібна фігура:

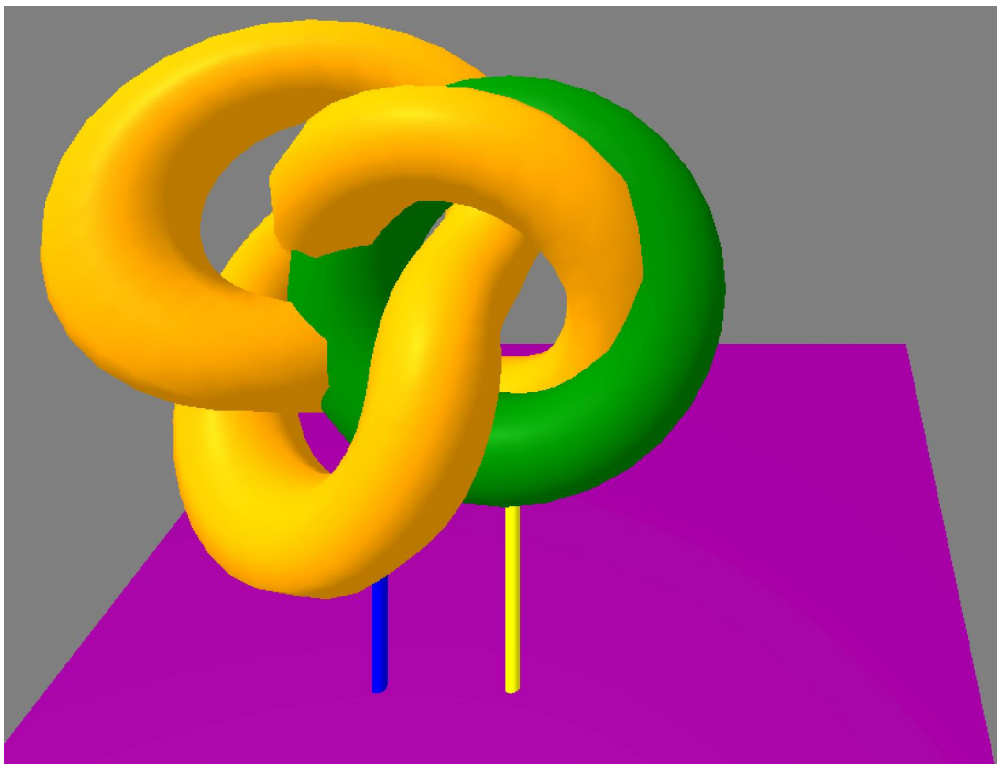
```
<a-torus-knot color="orange" radius="1.2"
              position="-3 1 -5"></a-torus-knot>
```



Досить цікава форма – без документації не розібратись.

2.3. Додавання тексту та анімації до об’єктів

Виконаємо невеликі зміни у індексному файлі в атрибуті обертання площини – зміна кута огляду на "-55 0 0" надає їй суттєво кращого вигляду.



Додамо до сцени площину, перпендикулярну попередній:

```
<a-plane width="9" height="2" position="3 1 -9"></a-plane>
```

За замовчанням її колір буде білим – достатньо зручний для того, щоб перед ним розмістити напис:

```
<a-text value="Welcome to browser's VR" color="black"
        width="10" position="-0.5 1 -6"></a-text>
```