Wstęp do grafiki komputerowej – projekty

Semestr zimowy 2024/2025

W tym semestrze zadania projektowe opierają się na programowaniu w Three.js/WebGL z użyciem bibliotek pomocniczych (physijs, cannon.js, babylon.js ….).

Oprócz hasłowych tematów podanych niżej, każdy z prowadzących laboratorium może zaproponować swój zestaw. Oczywiście wykonawca może zaproponować swój własny temat projektu, w podobnym stylu lub nawet całkiem odmienny (wtedy jednak rozsądna jest krótka konsultacja z prowadzącym).

W sieci można znaleźć sporo miejsc z przykładami projektów realizowanych w Three.js/WebGL, które mogą stać się inspiracją do własnych realizacji. Niektóre z prezentowanych projektów są całkiem skomplikowane, mogą jednak być inspiracją do prostszych wersji. Zrozumiałe też, że inspiracja, to nie kopiowanie plus np. kosmetyczne zmiany koloru.

Liczne, przykłady można znaleźć m.in. na stronach

- http://stemkoski.github.io/Three.js/ - niestety stara strona, ale z ciągle ciekawymi przykładami
- http://threejs.org/examples/ - oczywiście na tej stronie jest najwięcej przykładów, choć raczej ilustrujących tylko jeden mechanizm three.js
- https://uicookies.com/threejs-examples/ - zawiera 35 ciekawych i niezbyt złożonych projektów – myślę, że to jest bardzo dobre źródło.
- na stronie https://www.chromeexperiments.com/webgl zebrane są przykłady zrealizowane na Chrome, ale raczej w WebGL, a nie Three.js.
  Podobnie jest ze stroną http://www.hongkiat.com/blog/webgl-chrome-experiments/ zawierającą chyba 20 ciekawych i bardzo efektownych demonstracji, ale też w WebGL.

Tematy projektów mogą być realizowane jako jedno- lub dwuosobowe. Dwuosobowe powinny być bardziej rozbudowane, a poza tym obie osoby powinny być w projekcie zorientowane.

Zakończonemu i oddanemu projektowi powinien towarzyszyć krótki opis tekstowy zawierający informacje:

- Co realizuje projekt (można dodać instrukcję dla użytkownika, jeśli to potrzebne)
- Jaka jest struktura i zawartość kodu – z dokładnością do ważniejszych bibliotek i funkcji
- Jakie były źródła inspiracji – jeśli były, skąd pochodzą importowane modele, itp.

Chcę podkreślić, że opis projektu jest konieczny – choć nikt nie lubi tego robić. Może być zwarty.

Tematy przykładowych zadań projektowych.

1. Na podstawie mapy bitowej należy wygenerować siatkę mapy wysokości, pokolorować ją lub nałożyć teksturę i przedstawić w formie terenu nad którym można przelecieć samolotem. Należy dodać model samolotu (choćby bardzo uproszczony), samolotu, widziany jako *third person player*. Scenę można rozbudować o cień rzucany przez samolot, chmury, które mogą się pojawiać, mgłę, itp.
2. Przejście labiryntem przez kolejne pokoje, komnaty w pomieszczeniu. Labirynt powinien być generowany na podstawie wczytanej mapy. Projekt może być wykonany w formie fragmentu gry, w której możemy zbierać przedmioty, ewentualnie strzelać do przeciwnika.
3. Chodzący robot z możliwością prostego sterowania stylem chodzenia. Można wykorzystać na początek rozwiązanie z przykładu na stronie http://stemkoski.github.io/Three.js/ , ale to nie jest zbyt udany robot. Zdecydowanie lepszy byłby humanoid z Mixamo. Można wyposażyć robota w możliwość skoku i wskakiwania na prostopadłościenne przeszkody.
4. Jazda samochodem pomiędzy przeszkodami z prostopadłościanów. Samochód można zaimportować np. z Blendera, a prostopadłościany teksturować, tak, żeby wyglądały jak domy. Dobrze jest wykorzystać bibliotekę fizyki physijs lub cannon.js, wyposażyć samochód w zawieszenie (jak w jednym z przykładów physijs) i skręcane koła. Można też rozważyć inne efekty (np. kurz spod opon ruszającego samochodu).
5. Układanie kostek (sześcianów) jednych na drugich, tak jak klocków. Kostki można chwytać i przenosić za pomocą myszki. Do wykorzystania przykład z biblioteki physijs.

6. Model samolotu – własny lub ściągnięty z sieci, który startuje i/lub ląduje w nieskomplikowanym terenie. Bardzo uproszczony symulator lotu. Można dołączyć przełączanie widoku: z zewnątrz i z kabiny pilota.
7. Dźwig lub inny manipulator z połączonych elementów, który może przenosić obiekty. Może to być dźwig samojezdny, a elementy chwytać – co najprostsze – jakby magnetycznie. Można go jednak wyposażyć w sterowane kleszcze.
8. Fragment gry typu „strzelanka" FPP. Na pierwszym planie powinna znajdować się broń, która obraca się w zależności od tego w co celujemy. Obiekty, do których strzelamy mogą być bardzo proste. Jednak trafione powinny jakoś efektownie reagować – przewracać się, wybuchać, rozlatywać na części…
9. Gra w kręgle… wydaje się prosta, ale można ją skomplikować szczegółami dodającymi realizmu (otoczenie, tekstury, zderzenia kuli z kręglami, sposób przewracania się kręgli, itp.).
10. Fragment gry typu wyścigi samochodowe dla jednego lub dwóch zawodników. Mogą to być wyścigi po zamkniętym torze lub (co bardziej pracochłonne) po urozmaiconym terenie.
11. Zamodelowanie uproszczonej sceny „wyspa na morzu", do której możemy podpływać. Ważne jest to zamodelowanie krajobrazu, umieszczenie drzew, krzaków z teksturami (należy je zaimportować z sieci), ładne oświetlenie (typu wschód/zachód słońca).
12. Realizacja prostego modelu fal morskich w formie animowanej siatki. Wymaga użycia modelu matematycznego do animacji (np. w oparciu o klasyczną pracę Tessendorfa: http://graphics.ucsd.edu/courses/rendering/2005/jdewall/tessendorf.pdf, lub inne prostsze prace). Falującemu morzu może towarzyszyć unosząca się na powierzchni łódka. Alternatywnie można położyć nacisk na inne efekty: załamywanie się fal, rozbijanie się fal na brzegu (służę konsultacją).
13. Realizacja prostego systemu cząstek udającego ogień lub fontannę. Efekt cząstek powinien być wbudowany w jakąś scenę otoczoną skyboxem.
14. Wymodelowanie płomienia za pomocą cząstek, np. w efekcie płonącej świecy lub ogniska.
15. Wymodelowanie uproszczonej wersji mechanizmu, np. ruchomego modelu silnika tłokowego (bardzo uproszczonego).
16. Modelowanie efektów zimowych (np. do gry): padający śnieg pokrywa scenę równą warstwą.

17. Modelowanie padającego deszczu realizowanego za pomocą cząstek.
18. Wymodelowanie sceny z roślinami (drzewo, kilka drzew, kwiaty lub cokolwiek podobnego). Można wykorzystać bibliotekę www.snappytree.com, https://www.ibiblio.org/e-notes/webgl/tree.htm , lub biblioteki wykorzystujące L-systemy: https://github.com/nylki/lindenmayer .
19. Rozszerzona rzeczywistość. Na podstawie przykładu Stemkoskiego z kamerą internetową można zbudować układ, w którym np. do ustalonego znacznika dokleja się obiekt wygenerowany w komputerze. Jest też oddzielna strona Stemkoskiego poświęcona AR: https://stemkoski.github.io/AR-Examples/ i https://github.com/stemkoski/AR-Examples.
20. Wirtualna rzeczywistość. Coś czego nie robiliśmy. Można wykorzystać Three.js i WebXR do zbudowania sceny (właściwie dowolnej, np. górskiego terenu) i przedstawienia za pomocą okularów VR, choćby Google Cardboard.