Modeliranje i animacija modelom sustava masa i opruga

Aničić, Vanesa

SVEUČILIŠTE U RIJECI

TEHNIČKI FAKULTET

Preddiplomski sveučilišni studij strojarstva

Završni rad

MODELIRANJE I ANIMACIJA MODELOM SUSTAVA MASA I OPRUGA

Rijeka, rujan 2022.

Vanesa Aničić

0069086673

SVEUČILIŠTE U RIJECI

TEHNIČKI FAKULTET

Preddiplomski sveučilišni studij strojarstva

Završni rad

MODELIRANJE I ANIMACIJA MODELOM SUSTAVA MASA I OPRUGA

Mentor: izv. prof. dr. sc. Jerko Škifić

Rijeka, rujan 2022.

Vanesa Aničić

0069086673

Zadatak

SVEUČILIŠTE U RIJECI
TEHNIČKI FAKULTET
POVJERENSTVO ZA ZAVRŠNE ISPITE

Rijeka, 14. ožujka 2022.

Zavod: **Zavod za mehaniku fluida i računarsko inženjerstvo**
Predmet: **Računarske metode**
Grana: **2.15.04 mehanika fluida**

ZADATAK ZA ZAVRŠNI RAD

Pristupnik: **Vanesa Aničić (0069086673)**
Studij: Preddiplomski sveučilišni studij strojarstva

Zadatak: **Modeliranje i animacija modelom sustava masa i opruga / Mass spring system modelling and animation**

Opis zadatka:

Definirati model sustava masa i opruga. Izraditi računalni program kojem je cilj modeliranje gibanja objekata pod utjecajem vanjskih sila. Rezultat simulacije je potrebno prikazati kao animaciju, vodeći računa o mediju i osobinama materijala.
Polje znanstvenog područja: Temeljne tehničke znanosti
Grana znanstvenog područja: Mehanika fluida

Rad mora biti napisan prema Uputama za pisanje diplomskih / završnih radova koje su objavljene na mrežnim stranicama studija.

V. Aničić

Zadatak uručen pristupniku: 21. ožujka 2022.

Mentor:

Izv. prof. dr. sc. Jerko Škifić

Predsjednik povjerenstva za završni ispit:

Prof. dr. sc. Kristian Lenić

Rijeka, rujan 2022. Vanesa Aničić

0069086673

IZJAVA O AUTORSKOM RADU

Ja, Vanesa Aničić, izjavljujem da sam rad pod nazivom „Modeliranje i animacija modelom sustava masa i opruga“ izradila samostalno koristeći se navedenom literaturom i znanjem steknutim kroz dosadašnje obrazovanje. Rad sam izradila samostalno pod mentorstvom izv. prof. dr. sc. Jerka Škifića.

_____

Vanesa Aničić

ZAHVALA

Zahvaljujem se izv. prof. dr. sc. Jerku Škifiću na velikoj pomoći tijekom izrade ovog Završnog rada, kao i na njegovoj stručnosti i mentorstvu kojima me je vodio kroz pisanje Završnog rada. Zahvaljujem se svim kolegama i kolegicama s kojima sam provela preddiplomski studij i koji su ovo vrijeme studiranja uvelike uljepšali.

Najveće hvala mojoj obitelji, prijateljima i dečku koji su bili uz mene tijekom cijelog dosadašnjeg obrazovanja, bez čije bi potpore studiranje bilo daleko teže.

Sadržaj

UVOD

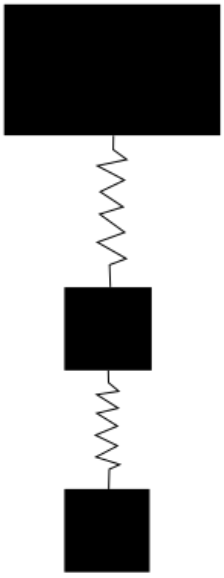
Računala su postala moćan alat za analizu podataka primjenom računalnih simulacija koje omogućavaju praćenje određenih pojava u realnom vremenu. Svaki računalni model podrazumijeva poznavanje zakonitosti fizike koji se odnose na zadani problem. Iz činjenice da su mogućnosti računala donekle ograničene, jasno je kako svaki matematički model korišten pri raznim simulacijama neće biti idealan. Ovaj je problem naglašen kod simulacija u realnom vremenu, no olakšavajući je faktor svakako naglasak na vizualnoj ispravnosti modela, a ne toliko na matematičkoj točnosti.

Postoji mnoštvo komercijalnih programa koji omogućavaju izvođenje relativno točnih simulacija, jedan od kojih je zasigurno Blender. Simulacija algi i podmorskog ekosustava izrazito je olakšala ponajprije svijet vizualnih medija kao sredstvo za izobrazbu i podučavanje, ali i svijet filma i animiranih filmova, koji primjenom novih načina 3D animiranja postaju sve realističniji prikaz stvarnosti.

U ovom radu objašnjavaju se razne metode koje se mogu koristiti prilikom kreiranja matematičkog modela samog sustava algi kao i njihove prednosti i nedostaci. Ukratko se predstavlja program za 3D vizualizaciju i animaciju Blender. Prikazuje se način modeliranja sustava primjenom metode masa i opruga, izrada animacije u Blenderu te programski kod kojim se model izrađivao.

1. MODEL MASA I OPRUGA

Samo tijelo alge bazira se na modelu masa i opruga. Model masa i opruga služi kao baza za modeliranje raznih problema iz inženjerske struke, ali i šire, primjerice kod modeliranja sportske opreme. Sustav je najjednostavnije objasniti kao masu ovješenu na slobodni kraj opruge, dok je drugi kraj opruge čvrsto vezan za podlogu. Ovakav se sustav može upotrebljavati za pronalaženje perioda titranja svakog objekta koji vrši harmonijsko gibanje. U ovom radu, tijelo alge predstavljeno je masama povezanim oprugama.



Slika 1.1 Sustav masa i opruga

Na slici (1.1) prikazan je sustav s tri mase i dvije opruge.

Sile koje djeluju u sustavu rezultat su djelovanja opruge te gravitacije, kao i težine ovješenog tijela.

Razmotrimo najprije osnovni sustav s jednom masom i jednom oprugom. Sila koje djeluje u osnovnom sustavu rezultat je djelovanja sile u opruzi, gravitacijske sile, kao i težine ovješenog tijela.

Izraz za gravitacijsku silu:

$$\overrightarrow{F_g} = m \cdot \vec{g} \text{ [N]} \tag{1.1}$$

gdje je:

m - masa objekta [kg]

g - gravitacijska konstanta koja iznosi $g = 9.81$ m/s^2

Sila u opruzi definira se kao:

$$\overrightarrow{F_S} = -k \cdot \Delta x = -k \cdot (x_2 - x_1) \text{ [N]} \tag{1.2}$$

gdje je:

k - koeficijent opruge [N/m]

Δx - je promjena položaja [m].

Ukoliko je koeficijent opruge k negativnog predznaka, znači da elastična sila djeluje suprotno od smjera sile koja izaziva produljenje ili skraćenje opruge, a ako je koeficijent opruge pozitivan, znači da elastična sila djeluje u smjeru sile koja izaziva produljenje ili skraćenje opruge.

Ukupna sila koja djeluje na osnovni model sa jednom masom može se dobiti kao:

$$\sum \vec{F} = m \cdot \vec{a} = \overrightarrow{F_S} - \overrightarrow{F_g} = k \cdot \Delta x - m \cdot \vec{g} \text{ [N]} \tag{1.3}$$

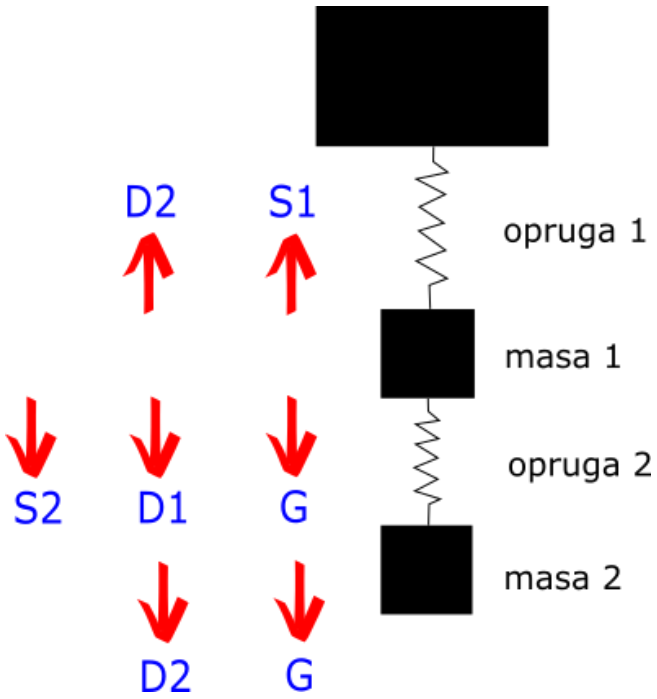
Ako pretpostavimo da obješeni objekt ne ubrzava, dobivamo slijedeću jednakost:

$$k \cdot \Delta x - m \cdot \vec{g} = 0 \tag{1.4}$$

1.1. Analiza sila koje djeluju na pojedinu masu u sustavu masa i opruga

Pogledamo li prethodno opisan sustav s dvije mase i dvije opruge, možemo postaviti jednadžbe koje opisuju djelovanje sila na svaku od tih masa.

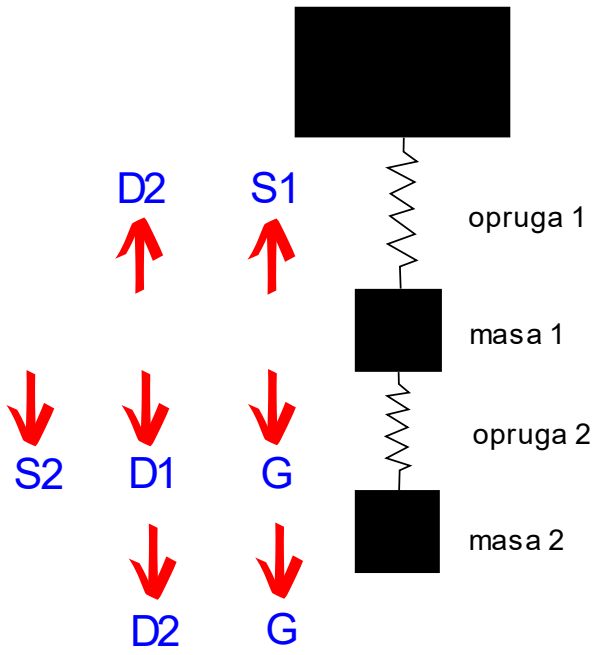
Skica sila kao i njihov smjer djelovanja prikazani su na slikama (1.2-1.3), gdje su S_1 i S_2 sile u opruzi, D_1 i D_2 sile prigušenja te G gravitacijska sila.



Slika 1.2 Sile koje djeluju na masu 1

Na masu 1 djeluju sila u opruzi 1, S_1, koja ju "vuče" za gore, gravitacijska sila mase 1, G, koja ju "vuče" za dolje te prigušenje, D_1, koja također djeluje preDma dolje. No, zbog mase 2, na masu 1 djeluje i sila u opruzi S_2 kao rezultat djelovanja mase 2, prigušenje D_2, koja djeluje prema gore.

Na masu 2 djeluje gravitacijska sila G, sila u opruzi S_2 koja djeluje prema dolje te prigušenje D_2, također sa smjerom djelovanja prema dolje (Slika 1.3).



Slika 1.3 Sile koje djeluju na masu 2

Sila u opruzi (označena kao S_1 i S_2), to jest elastična sila, sila je koja ima suprotan smjer djelovanja od smjera sile koja rasteže ili skraćuje elastično tijelo. Elastična sila je sila koja vraća tijelo u početni položaja nakon što prestane djelovanje vanjske sile koja ga razvlači. Za svako tijelo koje se opire promjeni oblika kaže se da ima elastičnu silu. S obzirom na to da se suprotstavlja deformaciji kako bi vratila tijelo u prvobitni položaj, to jest u ravnotežu, elastična sila se još naziva i regenerativna sila.

Prigušenje je jednako umnošku koeficijenta prigušenja d i brzine gibanja tijela v:

$$\overrightarrow{F_{\mathrm{d}}} = d \cdot \vec{v}\ [\mathrm{N}] \tag{1.1.1}$$

Rezultantna sila, uzmemo li u obzir sile koje djeluju na obje mase, imat će izraz:

$$\overrightarrow{F_{\mathrm{uk}}} = \overrightarrow{F_{\mathrm{s1}}} - \overrightarrow{F_{\mathrm{d1}}} - \overrightarrow{F_{s2}} + \overrightarrow{F_{\mathrm{d2}}} + m \cdot g\ [\mathrm{N}] \tag{1.1.2}$$

Isti princip primijenio bi se za sustav s proizvoljnim brojem masa i opruga.

2. DRUGI NEWTONOV ZAKON

Newtonov drugi zakon ili temeljni zakon gibanja kvanitificira promjene koje sila može proizvesti na tijelu koje se giba. Rezultantna sila može se jednostavno izraziti kao umnožak akceleracije i mase.

Izraz za drugi Newtonov zakon općenito glasi:

$$\vec{F} = m \cdot \vec{a} \text{ [N]} \tag{2.1}$$

gdje je:

a - akceleracija tijela [m/s^2]

m - masa tijela [kg]

iz čega se jednostavno može izraziti akceleracija:

$$\vec{a} = \frac{\vec{F}}{m} \text{ [m/s}^2\text{]} \tag{2.2}$$

Znamo kako je akceleracija derivacija brzine u vremenu, stoga integriranjem akceleracije možemo dobiti izraz za brzinu tijela:

$$\Delta\vec{v} = \int \vec{a}\, dt \tag{2.3}$$

Sada kada imamo brzinu, a poznato nam je kako je brzina derivacija puta u vremenu, možemo dobiti poziciju tijela kao:

$$\Delta s = \iint \vec{a}\, dt = \int \vec{v}\, dt \tag{2.4}$$

Ovaj nam zakon omogućava izradu modela sukladno novodobivenim pozicijama tijela. Postoje različite numeričke metode integracije modela u vremenu, o kojima će biti riječ kasnije.

2.1. Sustav običnih diferencijalnih jednadžbi

Kako bi prikazali gibanje alge, potrebno je odrediti brzinu i položaj svake kuglice tj. mase koja sačinjava tijelo alge.

Znamo da je brzina derivacija puta u vremenu:

$$\frac{dx}{dt} = \vec{v} \tag{2.1.1}$$

gdje je *x* put u metrima.

Dok je akceleracija derivacija brzine, tj. funkcija ovisna o putu i brzini:

$$\frac{d\vec{v}}{dt} = f(x, \overrightarrow{v}) \tag{2.1.2}$$

Za jednu masu sustava, to jest za jednu kuglicu, matrica početnih uvjeta (*ic*) imati će šest komponenti, gdje su najprije navedene komponente brzina po osima x, y i z, a zatim komponente akceleracije, također po osima x, y i z.

Općenito, sustav običnih diferencijalnih jednadžbi izgledat će:

$$\frac{d}{dt}\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} \overrightarrow{v_1} \\ \overrightarrow{v_2} \\ \vdots \\ \overrightarrow{v_n} \end{pmatrix} \tag{2.1.3}$$

Za sustav, kao u našem slučaju, s osam kuglica, matrica početnih uvjeta imati će 48 elemenata.

3. NUMERIČKE METODE RJEŠAVANJA OBIČNIH DIFERENCIJALNIH JEDNADŽBI

Nakon što smo postavili osnovne jednadžbe za sustav masa i opruga, potrebno je odabrati metodu numeričkog rješavanja običnih diferencijalnih jednadžbi, koja je ključna za dobivanje točne animacije.

Metode integracija u vremenu razvijene su kako bi se izbjeglo "ručno" rješavanje običnih diferencijalnih jednadžbi koje su osnova svake računalne simulacije. Zbog ograničenosti računalne tehnologije, sve metode integracije u vremenu samo su aproksimativne te služe za određivanje približnih rješenja, a ne egzaktnih. No, kako je u ovom radu važnija realistična animacija, preciznost same integracije nije u prvom planu.

U nastavku bit će opisane neke od metoda numeričkog rješavanja diferencijalnih jednadžbi.

3.1. Eulerova metoda integracije

Eulerova metoda integracije najstarija je i najjednostavnija metoda integracije.

Eksplicitna je metoda integracije jer se nova vrijednost računa tako da se određuje nagib pravca na početku intervala, koji se zatim preslikava duž cijelog intervala (slika 4.1.). [1]

Eulerova metoda pripada grupi Runge Kutta metoda.

Matematički izraz za izračun novih vrijednosti za određeni vremenski korak Δt:

$$y_{i+1} = y_i + y_i' \Delta t \tag{3.1.1}$$

$$y_i' = f(x_i, y_i) \tag{3.1.2}$$

gdje je $f(x_i, y_i)$ funkcija nagiba.

Na isti princip moguće je postaviti jednadžbe za brzinu i pomak:

$$v_{i+1} = v_i + a_i \Delta t \tag{3.1.3}$$

$$x_{i+1} = x_i + v_i \Delta t \tag{3.1.4}$$

Naravno, da bi dobili brzinu potrebno je integrirati akceleraciju u vremenu, a pomak, tj. novu poziciju, dobivamo integracijom brzine.

Metoda se može poboljšati tako da odaberemo manji vremenski korak, no to nije uvijek moguće jer se time usporava rješavanje modela.

3.2. Runge Kutta metoda

Runge Kutta metode koriste se za rješavanje početnog problema običnih diferencijalnih jednadžbi bez potrebe za derivacijama funkcija višeg reda.

Rješavanje običnih diferencijalnih jednadžbi Runge Kutta metodama izvodi se izračunavanjem niza točaka kao [1]:

$$y_{i+1} = y_i + \phi h \tag{3.2.1}$$

gdje je:

y_{i+1} – nova vrijednost

y_i – stara vrijednost

ϕ - koeficijent pravca

h – korak

3.2.1. Runge Kutta metoda četvrtog reda

Od svih Runge Kutta metoda najčešće se koristi Runge Kutta četvrtog reda ($n = 4$). [1]

Izraz za RK4 metodu glasi:

$$y_{i+1} = y_i + \frac{1}{6} \cdot (k_1 + 2k_2 + 2k_3 + k_4)h \tag{3.2.1.1}$$

gdje je k_i nagib pravca:

$$k_1 = f(x_i, y_i) \tag{3.2.1.2}$$

$$k_2 = f\left(x_\mathrm{i} + \frac{1}{2}h, y_\mathrm{i} + k_1\frac{1}{2}h\right) \tag{3.2.1.3}$$

$$k_3 = f\left(x_\mathrm{i} + \frac{1}{2}h, y_\mathrm{i} + k_2\frac{1}{2}h\right) \tag{3.2.1.4}$$

$$k_4 = f(x_\mathrm{i} + h, y_\mathrm{i} + k_3 h) \tag{3.2.1.5}$$

Nova vrijednost se računa pomoću već poznate vrijednosti, ali se uzimaju u obzir i neke međuvrijednosti funkcije [2]. Zbog toga je ova metoda nešto točnija od Eulerove metode.

3.3. Metoda srednje točke

Metoda srednje točke pripada grupi Runge Kutta metoda, točnije, spada u Runge Kutta metode drugog reda. Ova metoda uzima kao reprezentativni nagib onaj na sredini intervala, čime izraz glasi [1]:

$$y_{\mathrm{i}+1} = y_\mathrm{i} + \phi \cdot h \tag{3.3.1}$$

Za određivanje reprezentativnog nagiba, potrebno je najprije odrediti nagib na početku intervala:

$$y_\mathrm{i}' = f(x_\mathrm{i}, y_\mathrm{i}) \tag{3.3.2}$$

Koristeći izračunati nagib na početku intervala, možemo odrediti nagib na sredini intervala:

$$y_{\mathrm{i}+1/2} = y_\mathrm{i} + f(x_\mathrm{i}, y_\mathrm{i})\frac{h}{2} \tag{3.3.3}$$

Kada smo i to odredili, možemo odrediti nagib na sredini intervala:

$$\phi = f\left(x_{\mathrm{i}+1/2}, y_{\mathrm{i}+1/2}\right) \tag{3.3.4}$$

Konačno, sljedeću točku dobivamo kao:

$$y_{\mathrm{i}+1} = y_\mathrm{i} + \phi h = y_\mathrm{i} + f\left(x_{\mathrm{i}+1/2}, y_{\mathrm{i}+1/2}\right)h \tag{3.3.5}$$

3.4. Heunova metoda

Heunova metoda određuje nagib pravca kao linearnu kombinaciju nagiba pravca na početku i na kraju intervala. [1] Ova metoda također spada u grupu Runge Kutta metoda.

Kao i kod prethodnih metoda, najprije odredimo nagib na početku intervala:

$$y_{\mathrm{i}}' = f(x_{\mathrm{i}}, y_{\mathrm{i}}) \tag{3.4.1}$$

Potrebno je izračunati vrijednost točke na kraju intervala:

$$y_{\mathrm{i+1}}^{0} = y_{\mathrm{i}} + f(x_{\mathrm{i}}, y_{\mathrm{i}})h \tag{3.4.2}$$

Sada računamo nagib na kraju intervala:

$$y_{\mathrm{i+1}}' = f(x_{\mathrm{i+1}}, y_{\mathrm{i+1}}^{0}) \tag{3.4.3}$$

Kao reprezentativan nagib uzima se aritmetička sredina nagiba na početku i na kraju intervala:

$$\phi = \frac{y_{\mathrm{i}}' + y_{\mathrm{i+1}}'}{2} = \frac{f(x_{\mathrm{i}}, y_{\mathrm{i}}) + f(x_{\mathrm{i+1}}, y_{\mathrm{i+1}}^{0})}{2} \tag{3.4.4}$$

Napokon, računajući sa srednjim nagibom dobivenim pod (3.4.4), dolazimo do izraza za Heunovu metodu:

$$y_{\mathrm{i+1}} = y_{\mathrm{i}} + \phi h = y_{\mathrm{i}} + \frac{f(x_{\mathrm{i}}, y_{\mathrm{i}}) + f(x_{\mathrm{i+1}}, y_{\mathrm{i+1}}^{0})}{2} h \tag{3.4.5}$$

3.5. Integracija pomoću SciPy modula - funkcija odeint

Obične diferencijalne jednadžbe mogu se rješavati i koristeći modul SciPy koji sadrži programsku funkciju odeint, pogodnu za rješavanje početnog problema. Odeint se nalazi u podmodulu scipy.integrate.

Specifičnost rješavanja pomoću odeint leži u uvjetu da funkcija koja izračunava desnu stranu diferencijalne jednadžbe bude zapisana tako da joj najprije zapišemo nezavisnu varijablu te nakon nezavisne slijedi vektor zavisnih varijabli, u protivnom dobiveno rješenje neće biti ispravno. [1]

Zapis funkcije pri upotrebi odeint rješavača:

$$\frac{dy}{dx} = f \tag{3.5.1}$$

gdje je:

$$f = f(y, x) \tag{3.5.2}$$

4. IMPLEMENTACIJA MATEMATIČKOG MODELA

Za sustav masa i opruga gdje se broj masa, koeficijenti prigušenja i broj opruga može proizvoljno odabrati, možemo ispisati konačne jednadžbe sila koje djeluju na sustav masa i opruga. Također, s obzirom na to da se radi o tijelu uronjenom u vodu, moraju se uzeti u obzir i sile koje djeluju u fluidu.

4.1. Coriolisova sila

Coriolisova sila (F_C) je inercijska sila koja djeluje na sve čestice u rotirajućim sustavima koji se gibaju pod nekim kutom u odnosu na rotacijsku os sustava. [3]

Iako je utjecaj Coriolisove sile na sustav minimalan i zanemariv, odabrano je uvrstiti ga u razmatranje zbog fizikalne ispravnosti.

Izraz za Coriolisovu silu glasi:

$$\overrightarrow{F_C} = 2 \cdot \vec{v} \cdot sin(\varphi) \cdot \omega \text{ [N]} \tag{4.1.1}$$

gdje je:

v – brzina [m/s^2]

φ – geografski kut, za Republiku Hrvatsku je uzeto $\varphi = 45.2°$, što približno odgovara najzapadnijoj točki u RH u blizini mora.

ω – kutna brzina, koja iznosi $\omega = 7.29 \cdot 10^{-5}$ [rad/s] [4]

4.2. Sila otpora

Sila otpora (F_D) je sila koju uzrokuje gibanje tijela kroz neki fluid. Djeluje suprotno od smjera strujanja fluida. Da bi se sila otpora mogla generirati, tijelo mora biti u dodiru s fluidom. Ukoliko nema fluida, nema ni sile otpora. [5]

Izraz za silu otpora glasi:

$$\overrightarrow{F_D} = \frac{1}{2} \cdot C_D \cdot \rho \cdot \vec{v}^2 \cdot A \text{ [N]} \quad (4.2.1)$$

gdje je:

C_D - koeficijent otpora

ρ – gustoća fluida, to jest gustoća morske vode koja iznosi $\rho = 1026$ [kg/m^3]

v – brzina [m/s^2]

A – površina tijela [m^2]

Uzeli smo da je koeficijent otpora $C_D = 0.016$, a površina alge A = 7.82 m^2. [6]

4.3. Uzgon

Uzgon (F_u) je sila kojom fluid djeluje na tijelo uronjeno u taj fluid. Intenzitet kojeg sila uzgona proizvodi na tijelo u fluidu jednak je težini istisnutog fluida. U mirujućem fluidu, uzgon na neko tijelo djeluje prema gore, što je suprotno djelovanju gravitacijske sile. [5]

Izraz za silu uzgona je:

$$\vec{F}_u = \rho \cdot g \cdot V \text{ [N]} \quad (4.3.2)$$

gdje je:

ρ – gustoća fluida, to jest gustoća morske vode koja iznosi $\rho = 1026$ [kg/m^3]

v – brzina [m/s^2]

V – volumen tijela [m^3] kojega smo uzeli kao $V = 2.05 \cdot 10^{-2}$ [m^3] [6]

4.4. Jednadžba svih sila

Sada kada smo definirali sve vanjske sile koje djeluju na sustav, možemo ispisati konačnu jednadžbu svih sila koje djeluju na sustav, uzimajući u obzir da radimo s više masa i opruga te uzimajući u obzir djelovanja jedne mase na drugu.

Dolazimo do sljedeće jednadžbe:

$$\vec{F} = \vec{F}_{\mathrm{S[i]}} - \vec{F}_{\mathrm{d[i]}} - \vec{F}_{\mathrm{S[i+1]}} + \vec{F}_{\mathrm{d[i+1]}} + \vec{F}_{\mathrm{g}} - \vec{F}_{C\mathrm{[i+1]}} + \vec{F}_{\mathrm{D[i+1]}} + \vec{F}_{\mathrm{u[i+1]}}\ [\mathrm{N}] \qquad (4.4.1)$$

5. IMPLEMENTACIJA NUMERIČKOG MODELA

Za zadani problem možemo primijeniti numerički model kojim ćemo ga rješavati. Analizirat ćemo dobivene rezultate primjenom dva numerička modela: rješavanje početnog problema primjenom Scipy modula,primjenom Eulerove metode, Runge Kutta metode četvrtog reda, primjenom metode srednje točke te Heunove metode.

5.1. Primjena Scipy modula

Već spomenuta funkcija odeint ima sljedeću formu:

$$y = scipy.optimize.odeint(f, y0, x) \tag{5.1}$$

gdje je:

y – rješenje za pripadajuću x točku

f – funkcija koja opisuje običnu diferencijalnu jednadžbu: d*y*/d*x* = *f*(*y*,*x*)

y0 – početni uvjet

x – točka za koju se traži rješenje

U našem slučaju, sustav jednadžbi za koji se traži rješenje glasi:

$$\overrightarrow{F_{\mathrm{d}}} = d \cdot \vec{v} \text{ [N]} \tag{5.1.1}$$

$$\overrightarrow{F_{\mathrm{C}}} = 2 \cdot \vec{v} \cdot sin(\varphi) \cdot \omega \text{ [N]} \tag{5.1.2}$$

$$\overrightarrow{F_{\mathrm{D}}} = \frac{1}{2} \cdot C_{\mathrm{D}} \cdot \rho \cdot \vec{v}^2 \cdot A \text{ [N]} \tag{5.1.3}$$

$$\vec{F}_u = \rho \cdot g \cdot V \text{ [N]} \tag{5.1.4}$$

$$\vec{F} = \vec{F}_{\mathrm{S[i]}} - \vec{F}_{\mathrm{d[i]}} - \vec{F}_{\mathrm{S[i+1]}} + \vec{F}_{\mathrm{d[i+1]}} + \vec{F}_{\mathrm{g}} - \vec{F}_{C\mathrm{[i+1]}} + \vec{F}_{\mathrm{D[i+1]}} + \vec{F}_{\mathrm{u[i+1]}} \text{ [N]} \tag{5.1.5}$$

Tražimo brzinu ($\vec{v}$), koju dobivamo rješavanjem sustava jednadžbi. Izraz koji se u programskom kodu koristi za dobivanje rješenja je:

$$V_{\mathrm{x}} = itg.odeint(lambda\ Z, t: self.model(Z), Z0, t) \tag{5.1.6}$$

gdje je

V_x – brzina [m/s]

$Z0$ – matrica početnih uvjeta

self.model(Z) - funkcija koja vraća vektor svih evaluiranih vrijednosti svih jednadžbi sustava [1]

t – vremenski korak [s], matrica: $[0 \quad 0.1]$

5.2. Primjena Eulerove metode

Isti sustav jednadžbi predstavljen u potpoglavlju 5.1 riješen je koristeći i Eulerovu metodu.

Rješenje sustava dobivamo kao:

$$Z0 \mathrel{+}= self.model(Z0) \cdot h \tag{5.2.1}$$

gdje je:

h – interval definiran kao:

$$h = np.linspace(t[0], t[1], 30) \tag{5.2.2}$$

$t[0]$, $t[1]$ - komponente matrice vremena koje odgovaraju indeksima 0 i 1, a broj 30 predstavlja broj koraka.

5.3. Primjena Runge Kutta metode četvrtog reda

Sustav možemo riješiti i korištenjem Runge Kutta metode četvrtog reda. Tada izraz kojim dobivamo rješenje glasi:

$$Z0 \mathrel{+}= h \cdot (k_1 + 2\,k_2 + 2\,k_3 + k_4) \cdot \frac{1}{6} \tag{5.3.1}$$

gdje su k_i koeficijenti dobiveni sljedećim izrazima:

$$k_1 = self.model(Z0) \tag{5.3.2}$$

$$k_2 = self.model(Z0 + k1 \cdot \frac{h}{2}) \tag{5.3.3}$$

$$k_3 = self.model(Z0 + k2 \cdot \frac{h}{2}) \quad (5.3.4)$$

$$k_4 = self.model(Z0 + k3 \cdot h) \quad (5.3.5)$$

Kako je već navedeno, *h* je veličina intervala, a izračunava se kao:

$$h = np.linspace(t[0], t[1], 30) \quad (5.3.6)$$

gdje je 30 broj koraka koji se može proizvoljno odabrati.

Metoda se može dodatno poboljšati smanjenjem koraka *h*, no u tome smo ograničeni mogućnostima samog računala.

5.4. Primjena metode srednje točke

Nagib pravca računamo izrazom:

$$k_{\mathrm{m}} = self.model(Z0) + \frac{h}{2} \quad (5.4.1)$$

gdje je *h* veličina intervala:

$$h = np.linspace(t[0], t[1], 30) \quad (5.4.2)$$

Konačno, slijedi izraz za izračunavanje sustava pomoću metode srednje točke:

$$Z0 \mathrel{+}= k_{\mathrm{m}} \cdot h \quad (5.4.3)$$

5.5. Primjena Heunove metode

Interval *h* ponovno definiramo kao:

$$h = np.linspace(t[0], t[1], 30) \quad (5.5.1)$$

Nagib na početku intervala izrazimo kao:

$$k_1 = self.model(Z0) \quad (5.5.2)$$

Nagib na kraju intervala ima izraz:

$$k_2 = self.model(Z0 + k_1 * h) \quad (5.5.3)$$

Usrednjeni nagib dobivamo kao:

$$k = \frac{k_1 + k_2}{2} \tag{5.5.4}$$

Konačno, izraz za primijenjenu Heunovu metodu je:

$$Z0 \mathrel{+}= k * h \tag{5.5.5}$$

Za svaku od metoda bit će izrađena simulacija algi kako bi se prikazalo koja od njih daje najbolje rezultate.

6. UPORABA BLENDERA

6.1. O Blenderu

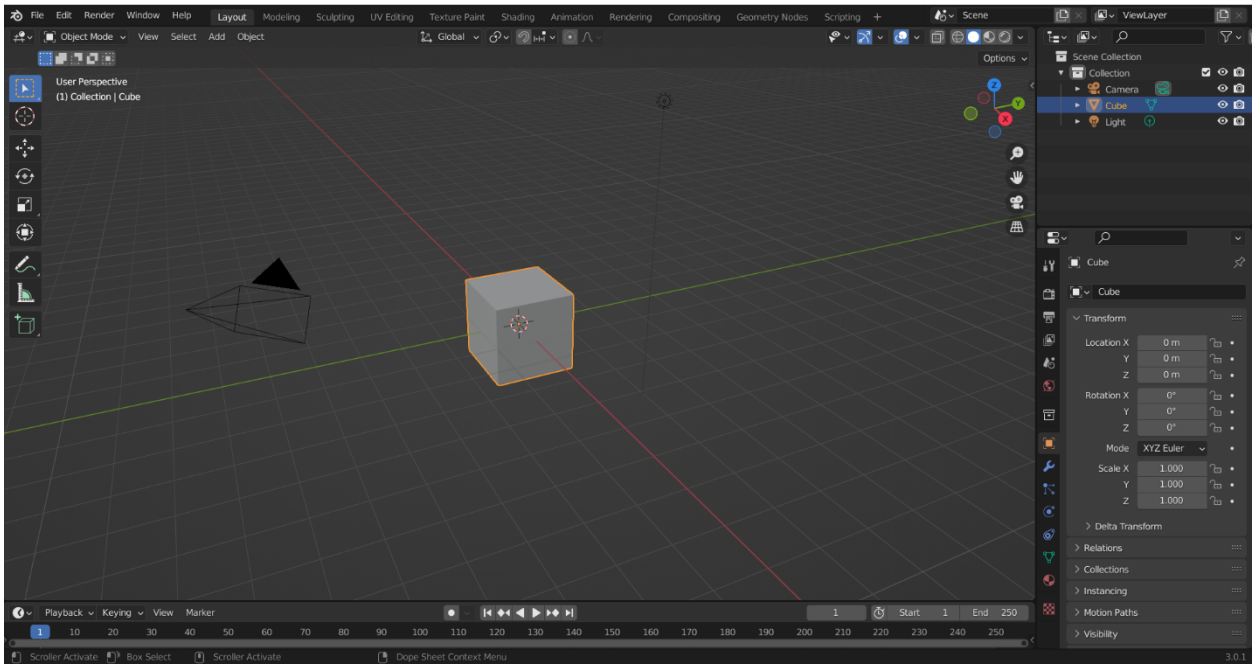
U ovom radu za animaciju i 3D prikaz modela sustava masa i opruga korišten je program Blender. Blender je besplatan program otvorenog koda koji se upotrebljava za modeliranje, izradu animacija, renderiranje, izradu simulacija pa čak i stvaranje i uređivanje video igara. [7] Vrlo korisna značajka ovog programa je i mogućnost pisanja vlastite skripte unutar Blenderovog API-ja (Application Programming Interface - programsko sučelje aplikacije) za Python, koja korisniku omogućava potpunu kontrolu nad programom kao i postizanje željenih rezultata.

6.2. Blenderovo sučelje

Kada se pokrene program, korisnika može odabrati između raznih vrsta blend datoteka u kojima može raditi, ovisno o projektu s kojime se bavi, a neke od njih su *General*, *2D Animation*, *Sculpting* i tako dalje (Slika 6.1). Za našu upotrebu najprikladniji je tip blend datoteke *General* (općenito).

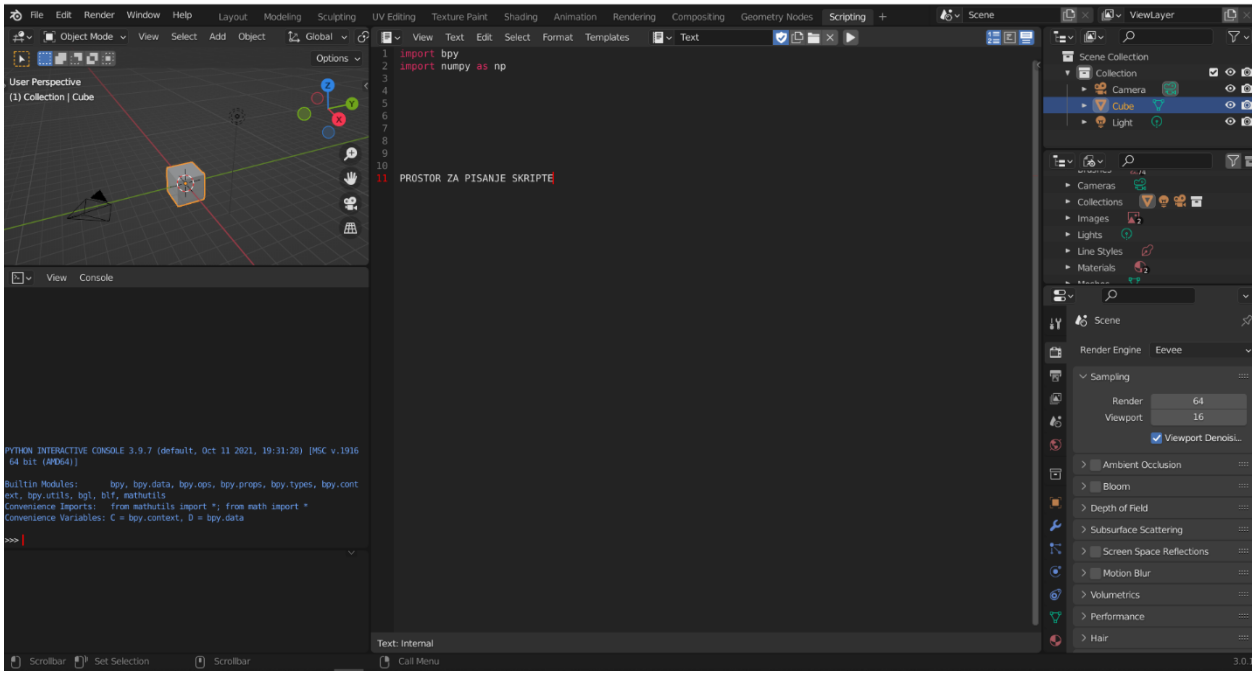
Slika 6.1 Blenderovo početno sučelje

Pri otvaranju svake nove blend datoteke u datoteci se nalaze tri standardna objekta: *primitive cube* (primitivna kocka), *camera* (kamera) i *lights* (svjetlo), kao što je vidljivo na slici 6.2. U ovom radu nisu nam potrebni spomenuti objekti jer svjetlo i kameru zadajemo zasebno, sukladno željenom izgledu scene, stoga ih možemo izbrisati.



Slika 6.2 Standardni objekti u Blenderu

Za ovaj rad važna je mogućnost pisanja Python skripte, koja je podržana u Blenderu te u tu svrhu na izbornoj traci možemo pronaći karticu *Scripting* (skriptiranje), klikom na koju možemo pisati Python kod (Slika 6.3).



Slika 6.3 Mjesto za pisanje Python koda u Blenderu

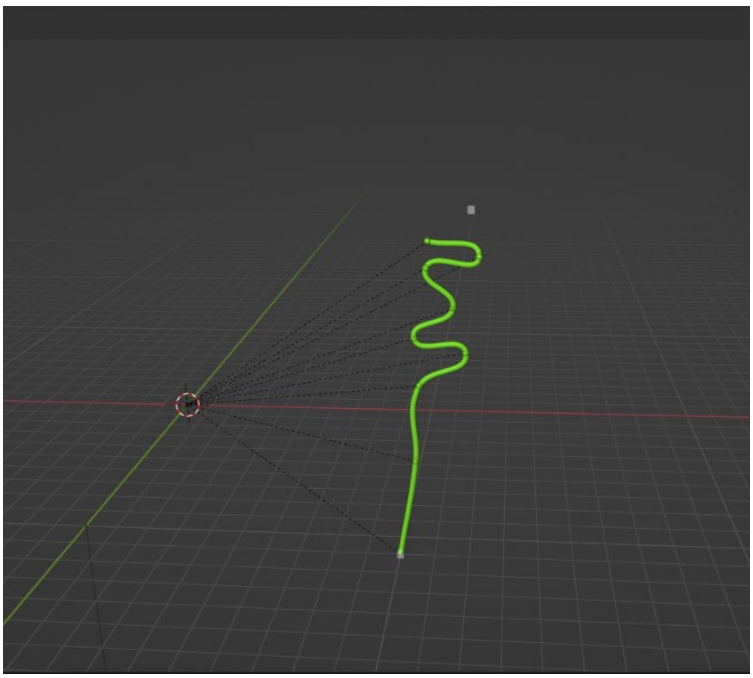
6.3. Izgled alge u Blenderu

Nakon što smo zadali sve sile koje djeluju na sustav masa i opruga te napisali kod koji modelira gibanje alge, možemo ju prikazati kao 3D model u Blenderu.

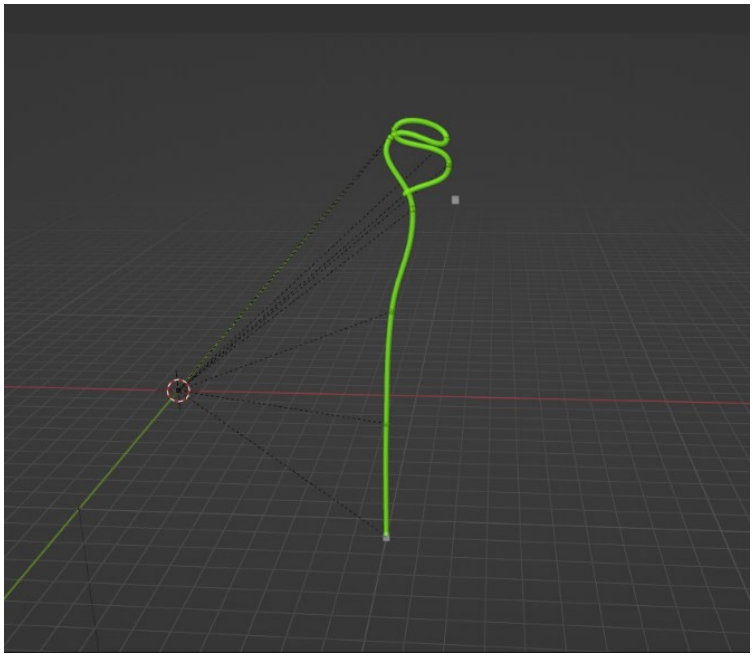
Boja same alge također je zadana programski.

Kada pokrenemo animaciju pritiskom na tipku *space*, alga se miče sukladno zadanom kodu.

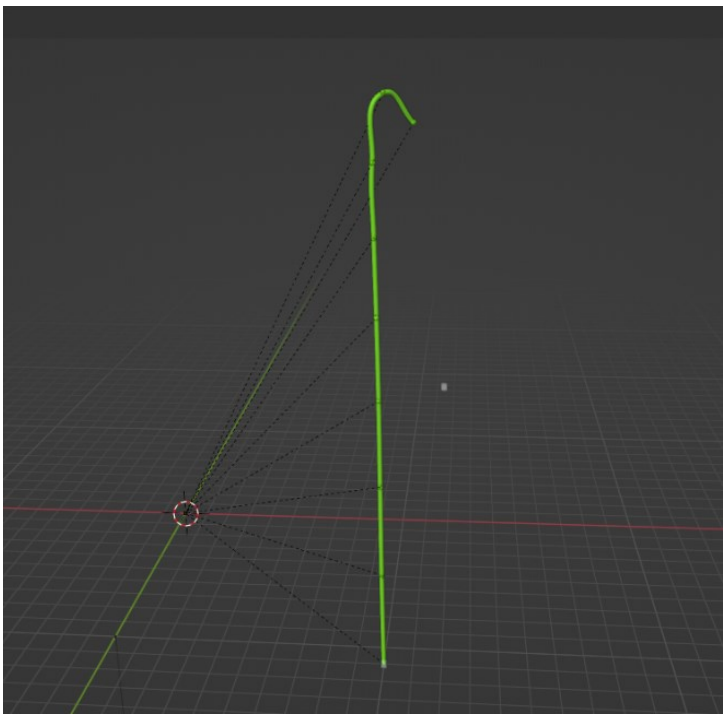
Na slikama (6.4 -6.5) vidljivo je gibanje alge do uspostave ravnotežnog stanja na slici 6.6.



Slika 6.4 Gibanje alge

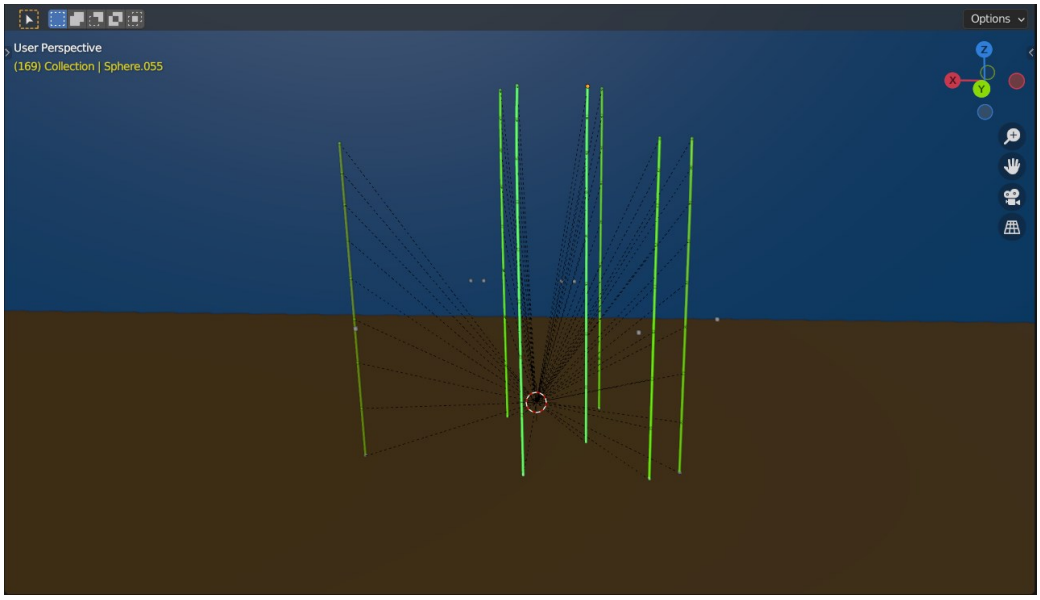


Slika 6.5 Gibanje alge

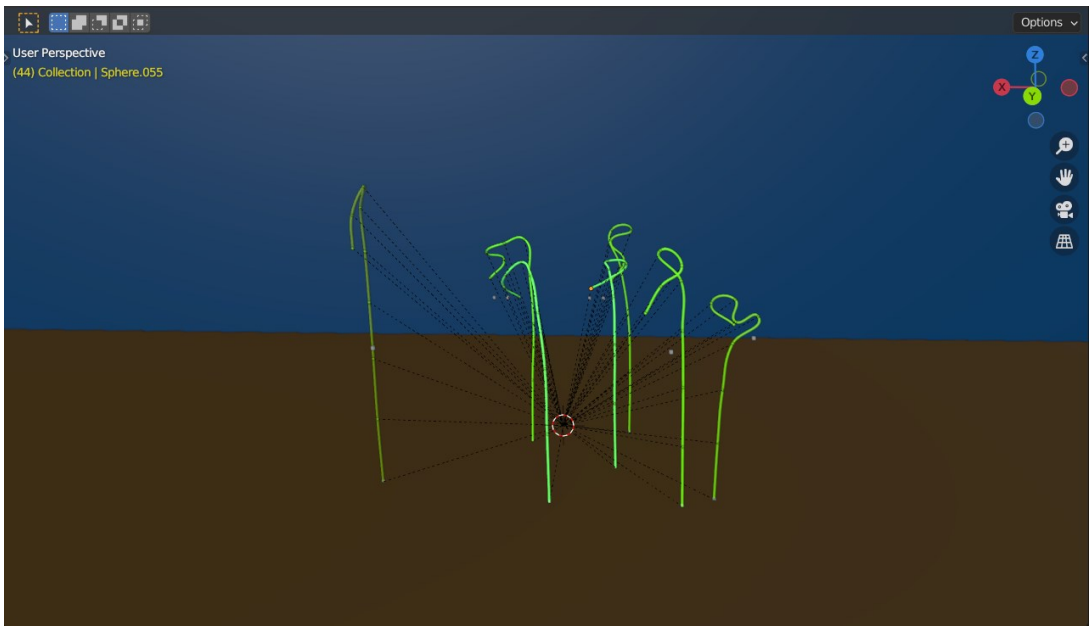


Slika 6.6 Gibanje alge

Nakon što je izražena jedna jedinka alge, možemo ih postaviti proizvoljno mnogo, ovisno o vlastitim željama. Primjer sa sedam jedinki alga dan je na slici 6.7, a na slici 6.8 može se vidjeti kako ta skupina algi izgleda u pokretu.



Slika 6.7 Skupina od sedam algi

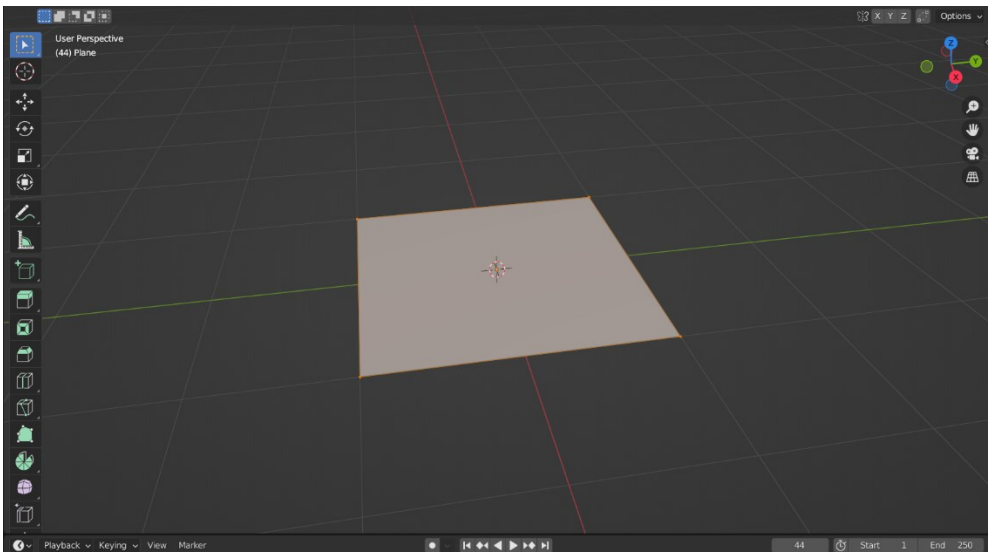


Slika 6.8 Alge u pokretu

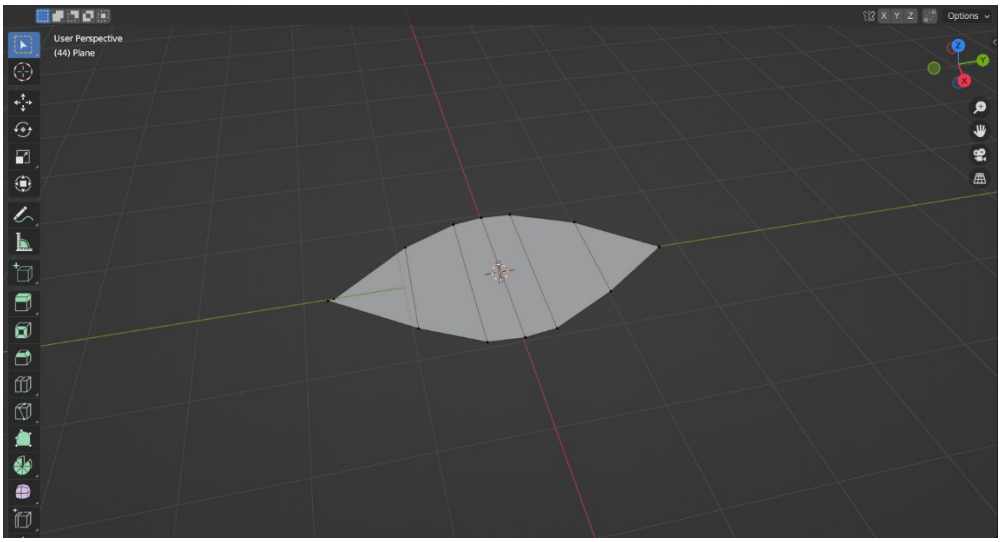
6.4. Modeliranje lišća

Modeliranje lišća u Blenderu može se vršiti na više načina, no ovdje se koristila jednostavna forma izrađena od ravnine (*Plane,* slika 6.9) koju smo transformirali i podijelili u segmente te oblikovali u željeni oblik lista.

Naredbom *Ctrl* + *R* omogućeno je brzo dijeljenje ravnine na željeni broj segmenata, koji se kasnije oblikuju u oblik lista, kao što je pokazano na slici 6.10.



Slika 6.9 Ravnina



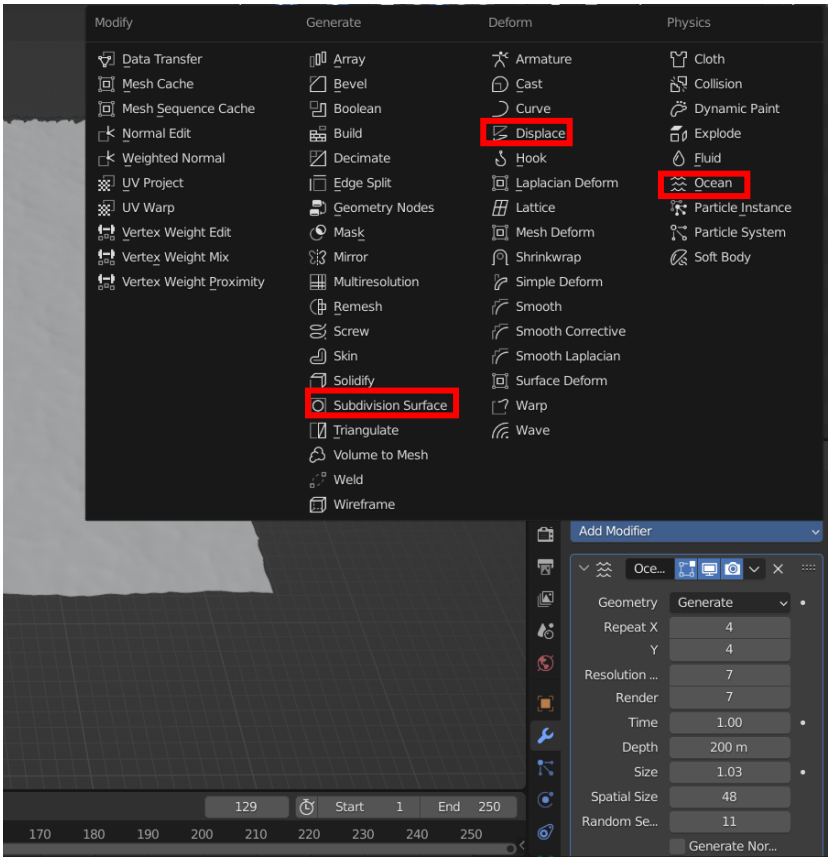
Slika 6.10 Grubi model lista

Boja lista automatski će se prilagoditi boji alge jednom kada spojimo (kratica *Ctrl + J*) ta dva objekta, stoga zasebno namještanje boje svakog lista nije potrebno.

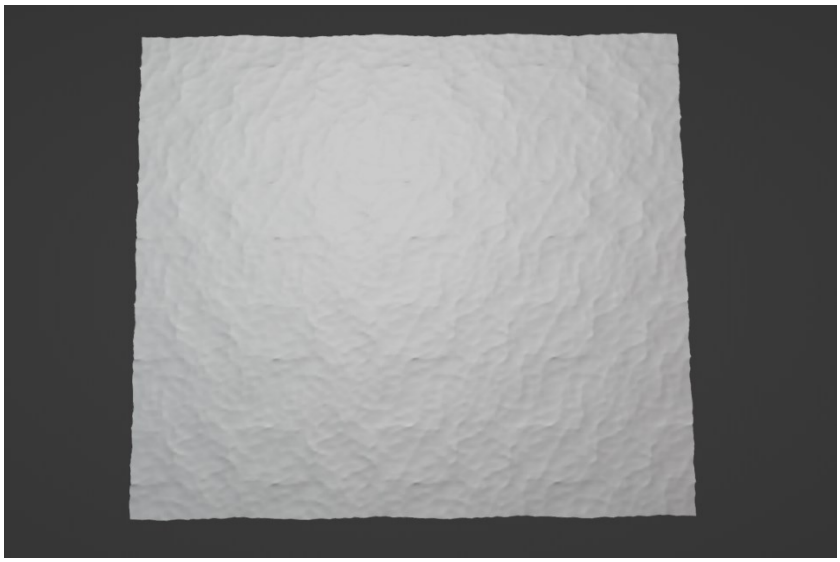
6.5. Modeliranje tla

Kako je morsko dno neravno i teksturirano, za prikaz morske podloge odabran je model sastavljen od ravnine na koju se dodaje željena tekstura.

Najprije se doda obična ravnina, na koju se zatim dodaje *Modifier* (Modifikator) (Slika 6.11). Mogući je izbor između mnoštva modifikatora, no ovdje je odabran modifikator koji svojom teksturom daje pjeskoviti i namreškani izgled, a to je modifikator *Ocean* (Slika 6.12).



Slika 6.11 Odabir modifikatora



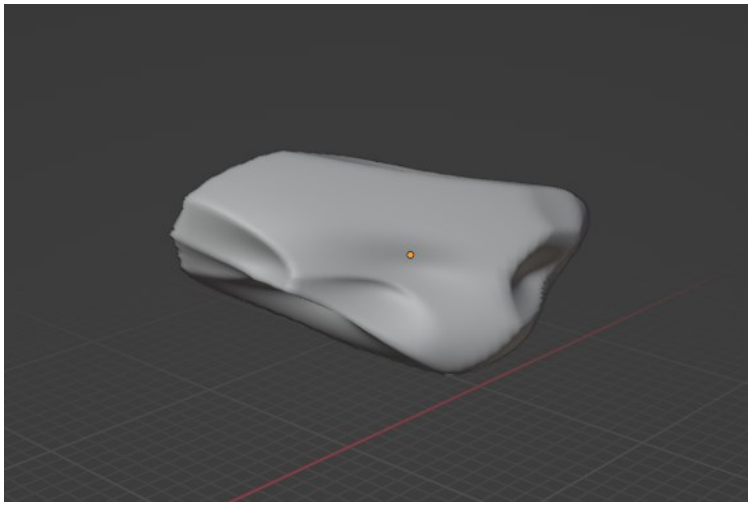
Slika 6.12 Ravnina nakon primjene modifikatora

Sada kada smo dobili osnovnu strukturu podloge, možemo na nju dodati ostale karakteristike značajne za morsko dno, kao što je kamenje.

Kamenje izrađujemo tako da najprije dodamo primitivnu kocku (*Cube*) kojoj dodamo dva modifikatora: *Subdivision surface* (podjela površine) i *Displace* (premještanje) (Slika 6.11).

Displace modifikator ima mogućnost biranja teksture objekta između već postojećih tekstura, ali i mogućnost kreiranja vlastite teksture. Za modeliranje kamenja korištena je tekstura *Voroni*.

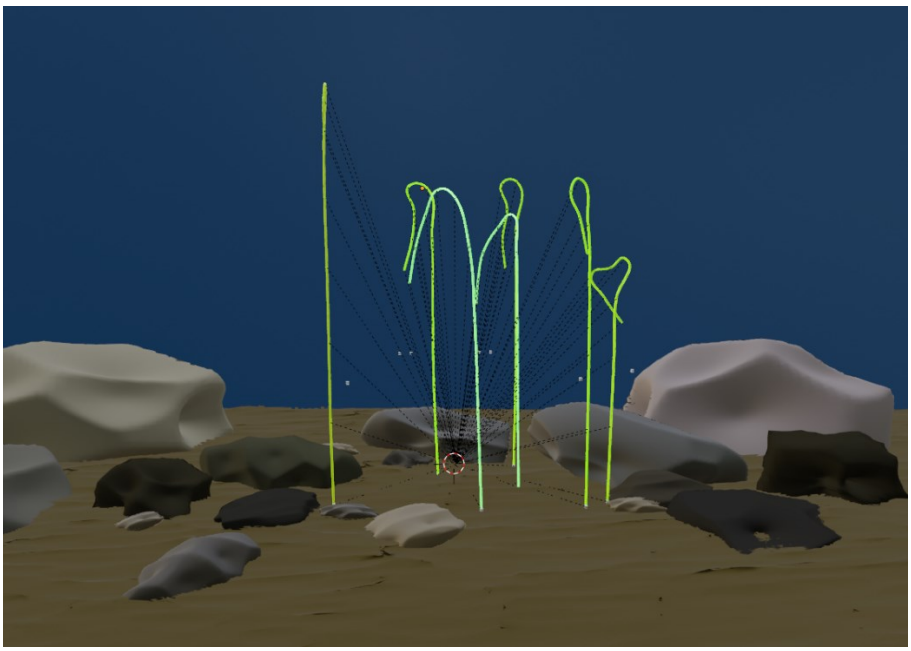
Modifikatorima oblikujemo kocku dok ne dobijemo oblik s kojim smo zadovoljni (Slika 6.13). Objekt sada možemo pozicionirati na željeno mjesto na podlozi (Slika 6.14).



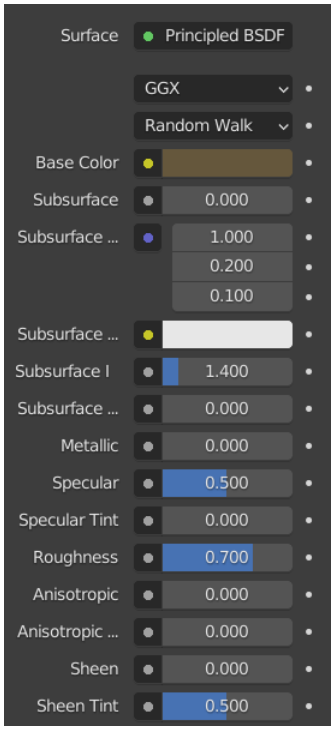
Slika 6.13 Model kamena

Boju kamenja, kao i podloge, namještamo preko kartice *Shading* (Sjenčanje), dodajući novi materijal svakom objektu. Možemo izabrati između efekta površine kamena, pa je tako u ovom radu korišteno više efekta površine: *Principled BSDF*, *Velvet BSDF*, *Toon BSDF* i *Diffuse BSDF*.

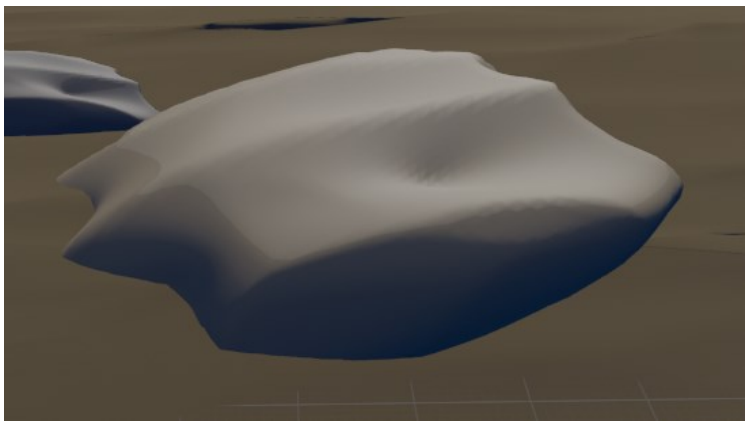
Na slici 6.15 prikazane su modificirane karakteristike kamena, kao što su *Base Color* (osnovna boja), *Specular* (zrcalnost), *Roughness* (hrapavost), *Sheen tint* (nijansa sjaja). Navedene efekte koristili smo kako bi dobili izgled kamena prikazanog na slici 6.16.



Slika 6.14 Model okoline



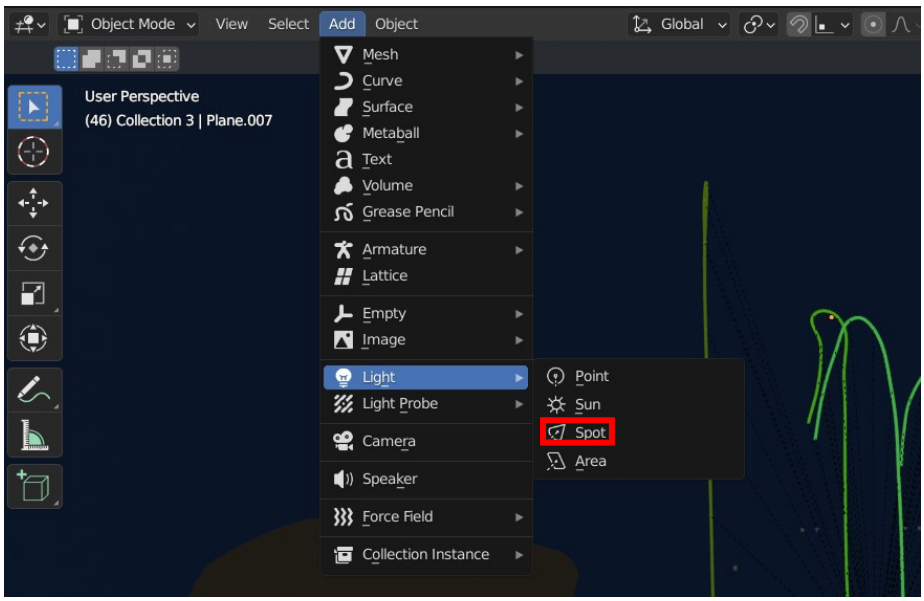
Slika 6.15 Postavljanje materijala kamena



Slika 6.16 Izgled kamena s postavljenim materijalom

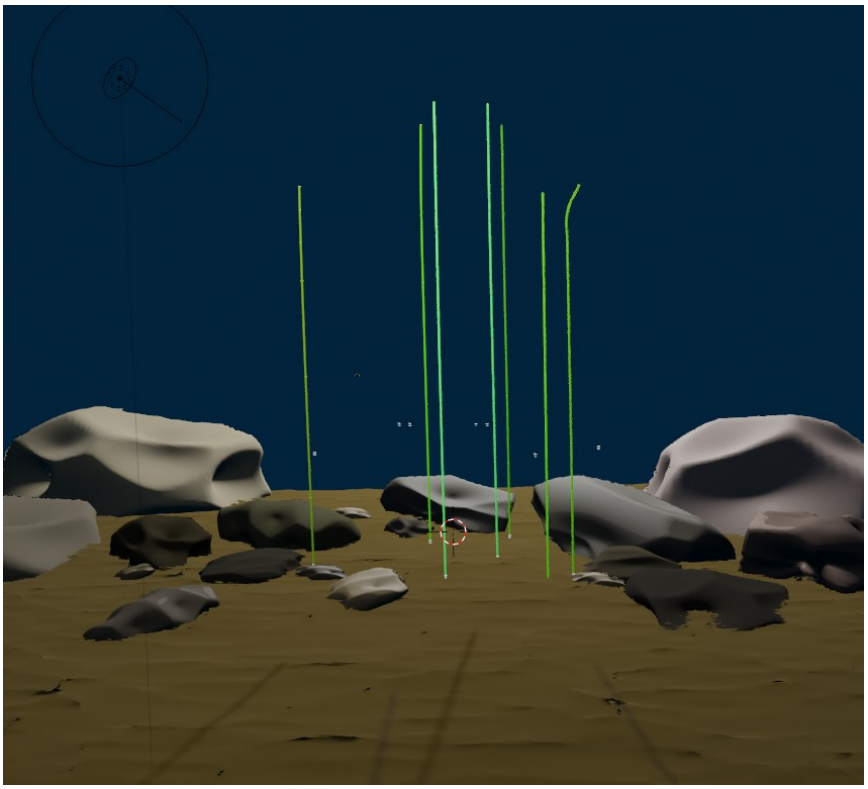
6.6. Postavljanje svjetla

Osvjetljenje scene jedan je od ključnih elemenata koji animaciji daju dubinu i realističan izgled. U Blenderu se svjetlo može zadati kao *Poin*t (Točka), *Sun* (Sunce), *Spot* (Mrlja) ili *Area* (Površina) (Slika 6.17). Odabran je objekt *Spot* te je postavljen iznad modela algi, kako bi sjene koje stvara na okolinu bile slične onima na morskom dnu. Iako bi možda očit odabir bio *Sun*, on daje preveliku koncentraciju svjetla, dok *Spot* daje dojam prividnog probijanja sunčevih zraka kroz more.



Slika 6.17 Dodavanje svjetla

Jačina svjetla može se podesiti desnim klikom na dodani objekt svjetla te tako podešavamo razinu osvjetljenja okoline. Prikaz okoline s dodanim svjetlom vidljiv je na slici 6.18.



Slika 6.18 Okolina nakon dodavanja svjetla

Osim ručno, svjetlo možemo zadati i putem koda.

Kod za dodavanje svjetla 1 glasi:

$$bpy.ops.object.light\_add(type = 'SPOT', radius = 1.35, location = (-80.091, -357.5, 584.9), rotation = (0.572586, -0.192863, 0.244160), scale = (1, 1, 1)) \quad (6.6.1)$$

Dodavanje svjetla 2:

$$bpy.ops.object.light\_add(type = 'SPOT', radius = 3.1, location = (33.871, -351.1, 647.46), rotation = (0.563589, -0.061301, 0.364262), scale = (1, 1, 1)) \quad (6.6.2)$$

Dodavanje svjetla 3:

$$bpy.ops.object.light\_add(type = 'SPOT', radius = 40, location = (123.97, 404.28, 330.54), rotation = (-1.094994, 0.679300, -0.106593), scale = (1, 1, 1))$$

(6.6.3)

gdje je:

bpy.ops.object - poziv operatora

camera_add – dodaje objekt svjetla

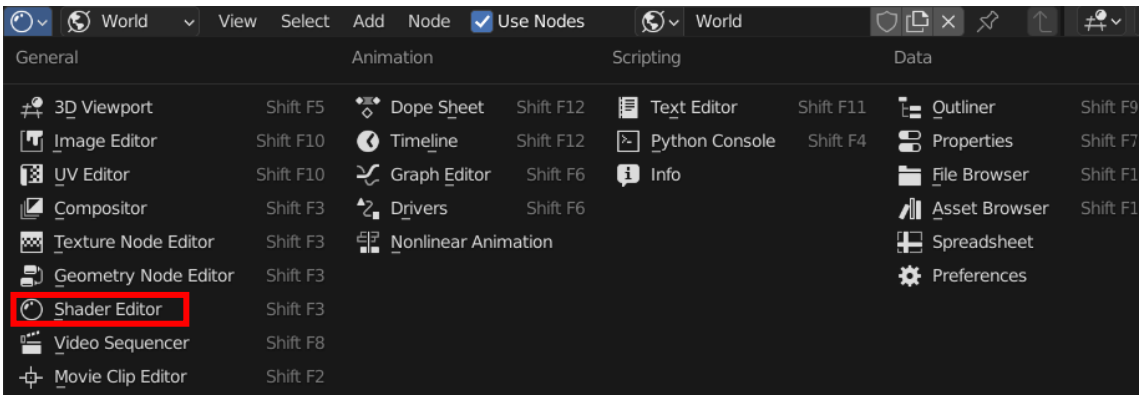
location – lokacija svjetla po x, y i z osi u metrima

rotation – rotacija svjetla oko osi x, y i z u radijanima

6.7. Modeliranje vode

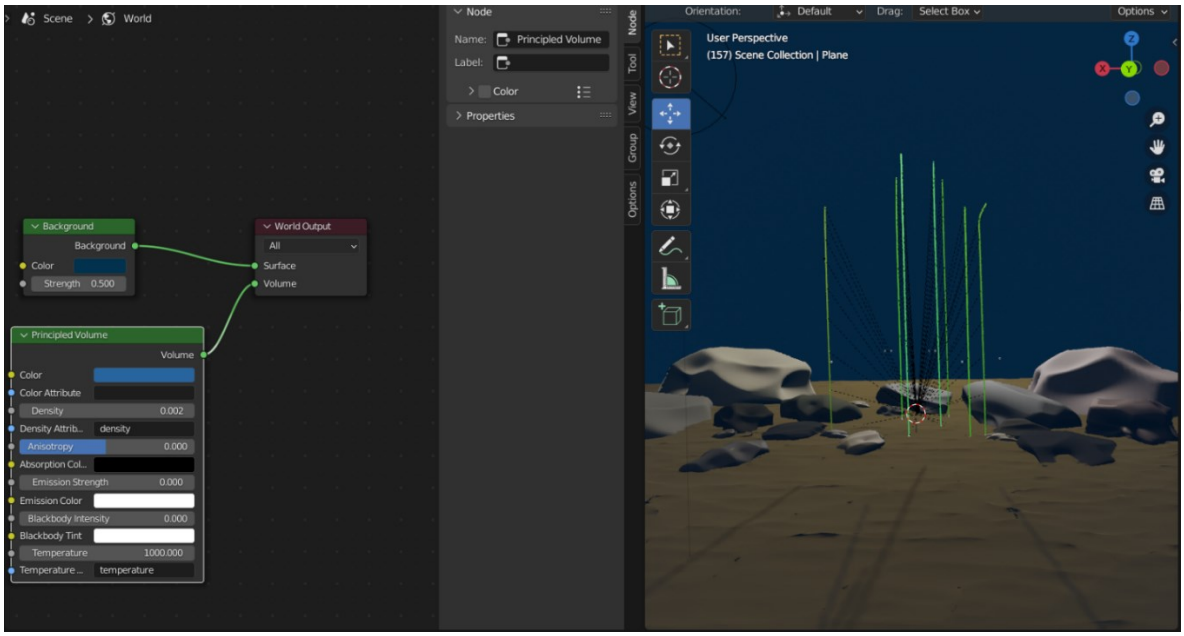
Kako bi sama scena bila još realnija i sličnija izgledu podmorja, potrebno je dodati vodu u kojoj alge stanuju. Vodu modeliramo tako da okruženju (*World*) dodamo *Principled Volume* stavku kojom možemo regulirati boju, prozirnost i izgled okruženja u kojem radimo. Navedeno vršimo preko takozvanih čvorova (*Nodes*) koje su dio *Node Editor* uređivača. Materijali i izgled objekta u Blenderu mogu se zadavati preko pripadajućih kartica koje za to služe, no kako bi se ubrzao proces kreiranja modela i scena, Blender ima mogućnost povezivanja osobina materijala preko čvorova. Svaki čvor vrši neku operaciju nad materijalom, mijenjajući kako će izgledati kada ga primijenimo na objekt. [8] Čvorovima se mogu postići kompleksniji i detaljniji materijali, stoga je poznavanje njihove uporabe vrlo praktično i korisno.

Node Editor kojim smo u ovom radu dodali efekt podmorja, uključujemo odabirom *Shader Editor* iz padajućeg izbornika (Slika 6.19).

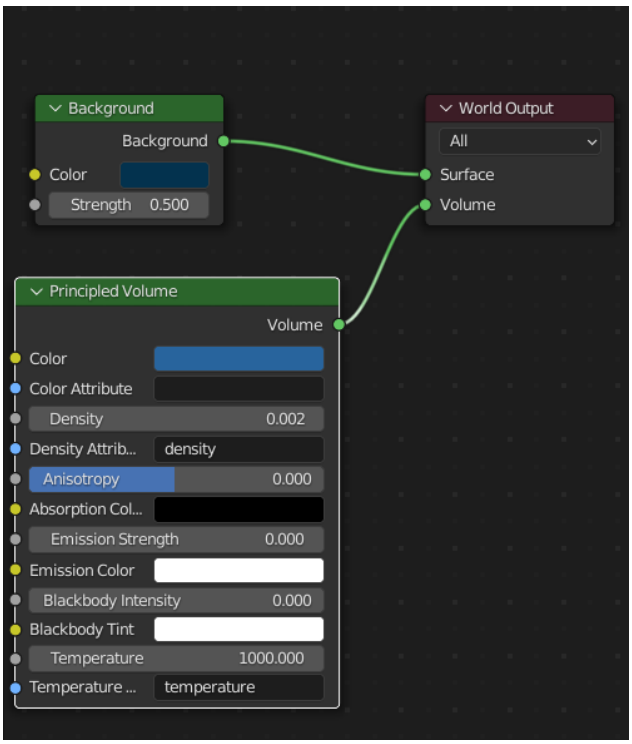


Slika 6.19 Odabir uređivača čvorova

Sada možemo povezati čvorove koji spajaju *Principled Volume* i *Volume*, kako bi dobili željeni efekt, što je prikazano na slikama (6.20-21). Također, možemo podesiti prozirnost tj. gustoću medija, boju, temperaturu te još niz značajki koje nisu korištene u ovom radu.

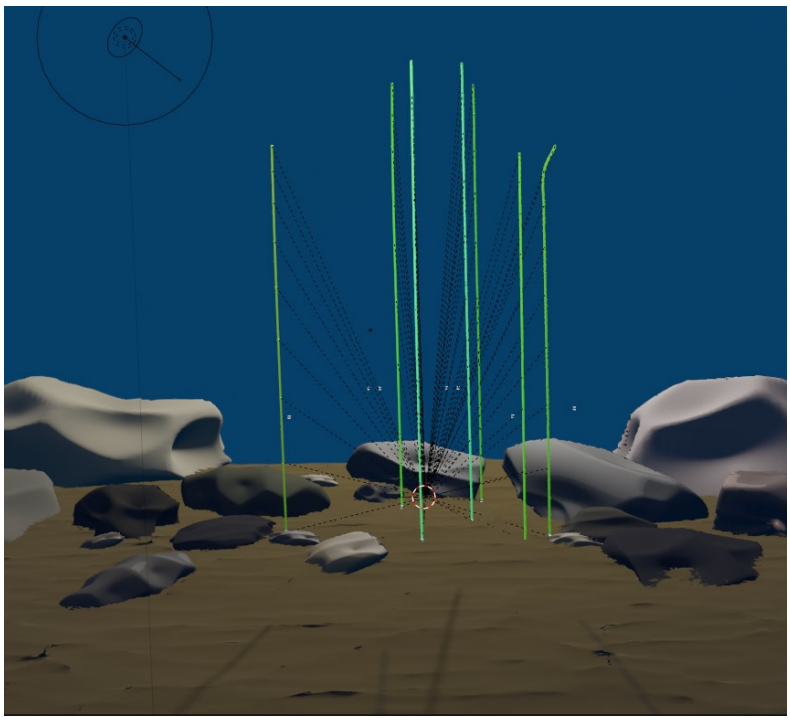


Slika 6.20 Node Editor i okruženje u Blenderu



Slika 6.21 Čvorovi za dobivanje efekta vode

Sada smo dobili plavkastu nijansu koja okružuje naš model te tako simulira podvodnu scenu (Slika 6.22).

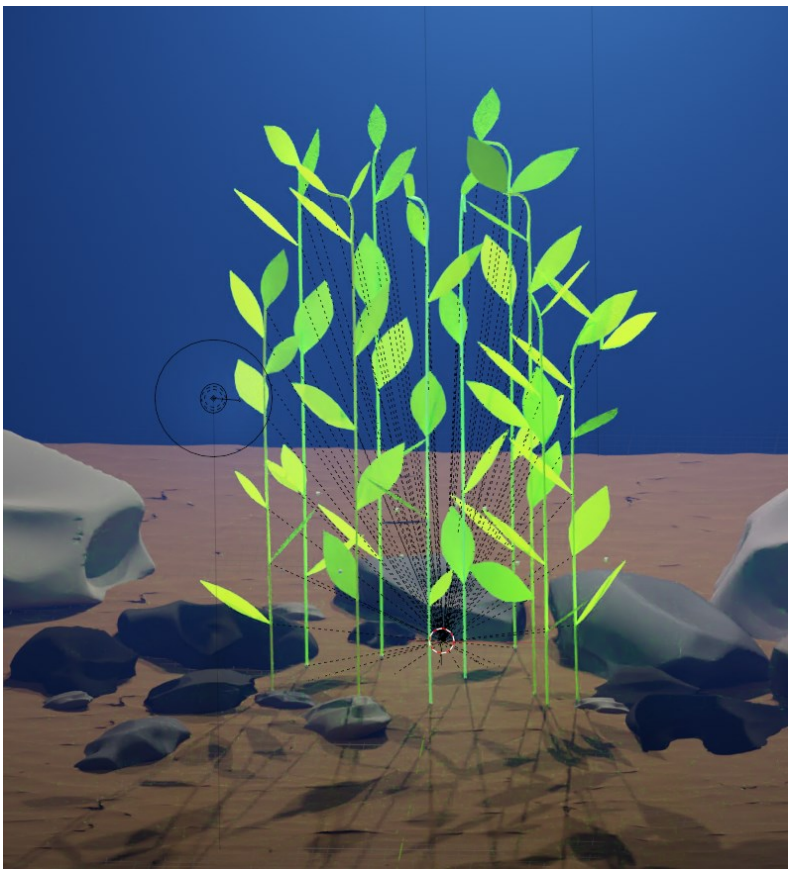


Slika 6.22 Okolina nakon primjene uređivača čvorova

6.8. Dodavanje lišća

Lišće smo modelirali na način kako je prikazano u potpoglavlju 6.4. Kako bi jednostavno izradili željeni broj listova, početni model možemo kopirati željeni broj puta. Kopiranjem objekta lista stvaramo identičnu kopiju jednog lista, a kako je u prirodi rijetko slučaj da su listovi identični, svaku kopiju lista možemo transformirati korištenjem alata *Transform* (transformiranje), *Scale* (skaliranje), *Rotate* (zakretanje). Na slikama 6.23 i 6.24 vidljiv je model algi s umetnutim lišćem.

Slika 6.23 Alge s dodanim lišćem (Eevee renderer)



Slika 6.24 Alga s dodanim lišćem (Cycles renderer)

6.9. Postavljanje kamere

Nakon što smo modelirali okruženje koje zadovoljava naše potrebe te dodali svjetlo i lišće na model alge, potrebno je postaviti kameru koja će prikazivati točno ono što želimo da nam animacija sadrži.

Kameru možemo dodati ručno kao objekt u Blenderu preko Add - Camera, ali možemo ju zadati i preko koda, kao što je slučaj ovdje.

Kod kojim dodajemo kameru glasi:

$$bpy.ops.object.camera\_add(location = (10, 450, 200), rotation = (1.5, 0.013963, 3.15)) \quad (6.9.1)$$

gdje je:

bpy.ops.object - poziv operatora

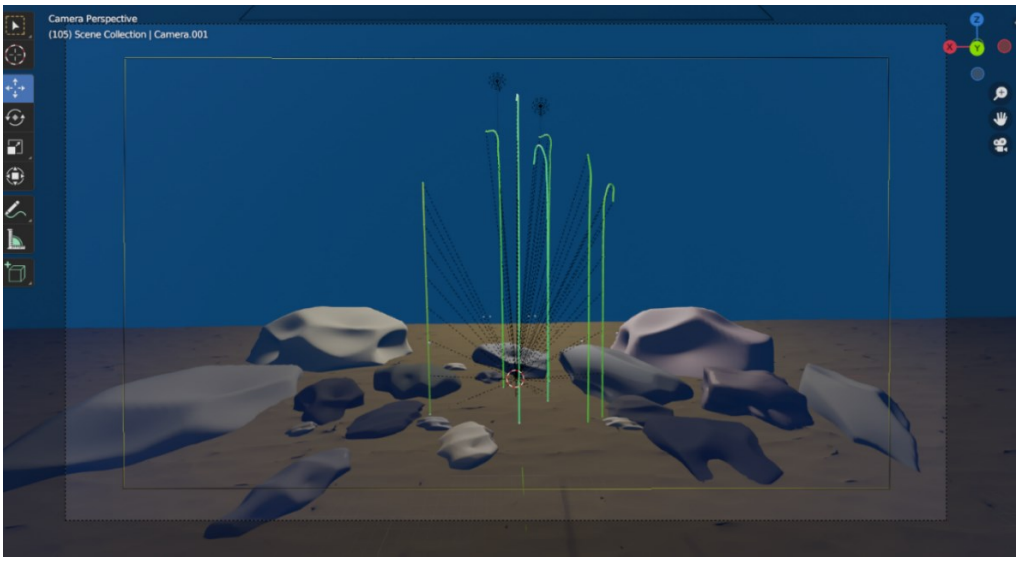
camera_add – dodaje objekt kamere

location – lokacija kamere po x, y i z osi u metrima

rotation – rotacija kamere oko osi x, y i z u radijanima

Kako bi točno pozicionirali kameru isprobavano je više pozicija po koordinatnim osima kao i više rotacija oko koordinatnih osi, dok naposljetku nije dobivena pozicija kamere koja zadovoljava i koja obuhvaća scenu na željeni način.

Pritiskom na ikonicu kamere na desnoj strani Blenderovog okruženja može se pogledati kako kamera „vidi“ scenu, te tako odrediti jesu li potrebne preinake u lokaciji ili rotaciji kamere (Slika 6.25).



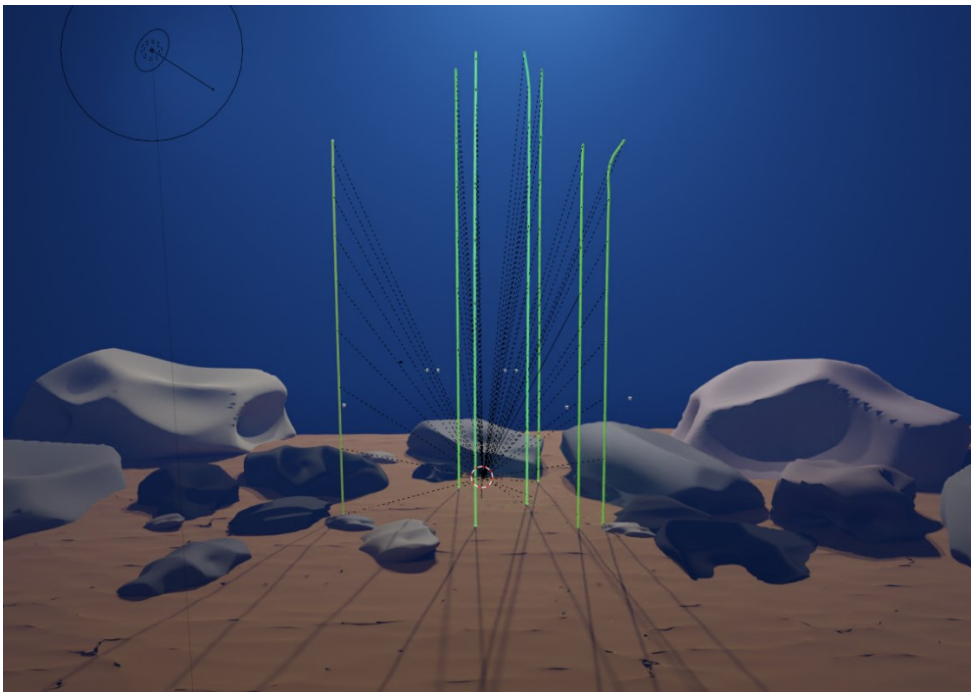
Slika 6.25 Pogled iz kamere

7. RENDERING

Rendering (prikazivanje) je proces izvoza projekta na kojem radimo koji može biti 2D dizajn ili 3D model. Omogućuje da izrađeni projekt izgleda foto-realistično. [9] Blender ima četiri mogućnosti pri odabiru izvršitelja za rendering: *Workbench*, *Eevee*, *Cycles* i mogućnost korištenja third-party programa za renderiranje.

Workbench je prikladan za jednostavnije zadatke, *Eevee* vrši render u realnom vremenu, dok *Cycles* donosi renderiranje najviše kvalitete i robusnost. [10]

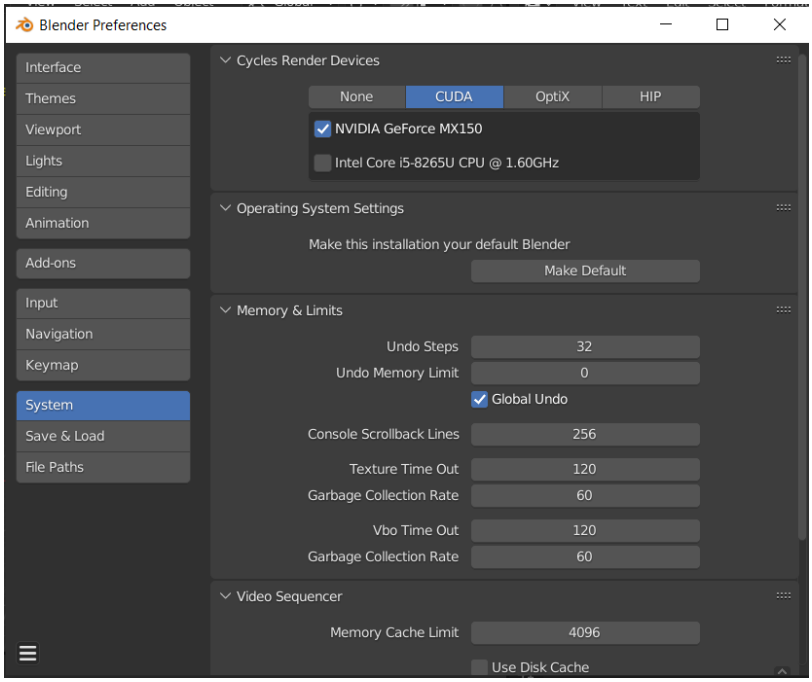
Svi prikazi modela i scene rezultat su *Eevee* izvršitelja renderinga, koji je brži i za računalo lakši od *Cycles* izvršitelja, no *Cycles* daje daleko realističniji izgled scene i modela, stoga je on korišten za izradu jedne animacije, a *Eevee* je korišten za izradu animacija za usporedbu između korištenih metoda rješavanja sustava. Izgled scene i modela bez lišća, grubo renderiran *Cycles* izvršiteljem prikazan je na slici 7.1.



Slika 7.1 Grubi Cycles render

Cycles renderer pogodan je jer, kao što je već rečeno, daje najrealističnije rezultate zbog toga što iskorištava potencijal grafičkih kartica pomoću kojih vrši rendering. Kako bi se uopće mogla

koristiti snaga Cycled renderera i grafičke kartice, u Blenderu je potrebno omogućiti renderiranje preko GPU-a (*Graphic Processing Unit*). Navedeno se omogućava ulaskom u *Blender – Edit – Preferences – System*, gdje se može odabrati *Cycles Render Devices*. Za potrebe ovog rada odabrana je CUDA koja je podržana na Windows i Linux uređajima te zahtijeva Nvidia grafičku karticu. [11] Izbornik za odabir GPU renderinga prikazan je na slici 7.2.



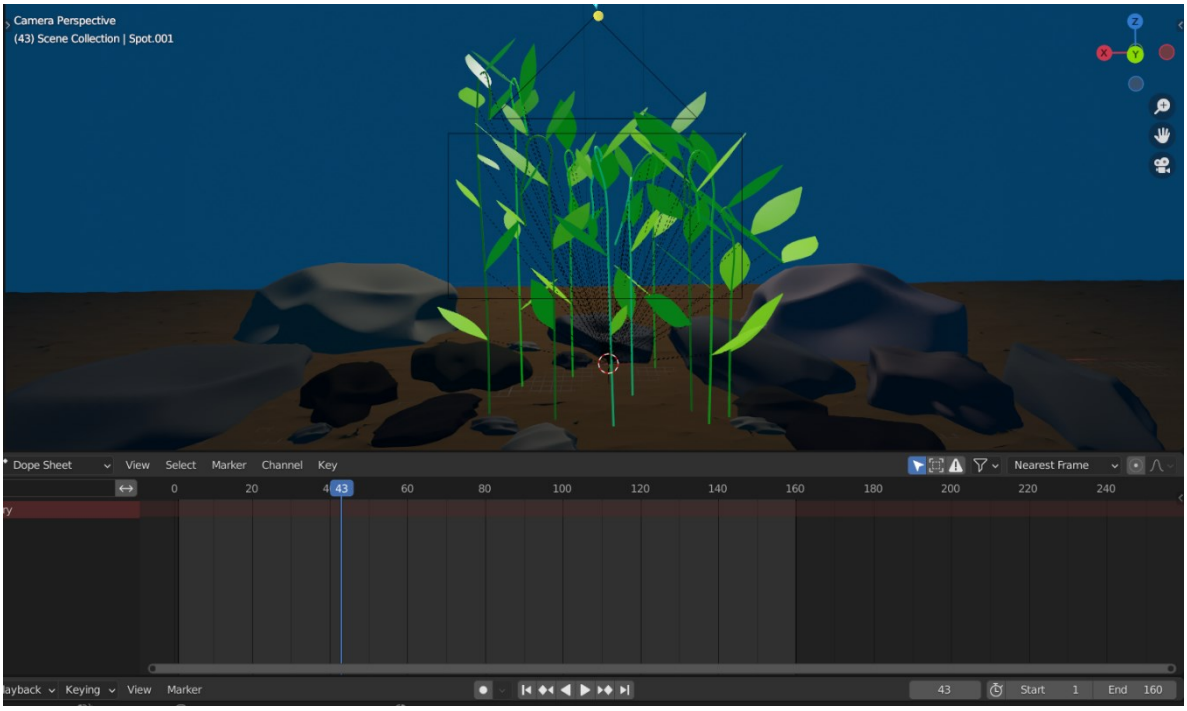
Slika 7.2 Odabir GPU renderinga

8. IZRADA ANIMACIJE

Izrada animacije vrši se u za to predviđenoj kartici u Blenderu - *Animation*. Kako smo već prije namjestili kameru, dovoljno je samo odabrati početni i krajnji *frame* za izradu animacije.

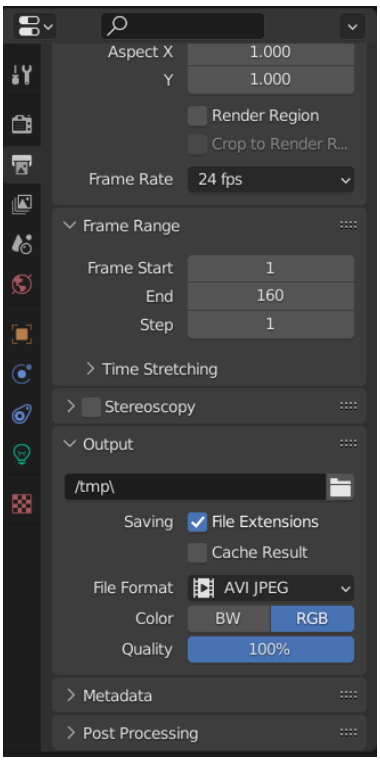
Za ovaj je rad odabrano 160 frame-ova jer tako dobijemo dovoljno dugu animaciju u kojoj se prikaže cijelo gibanje alge do uspostavljanja ravnoteže.

Prozor za odabir i manipulaciju frame-ovima prikazan je na slici 8.1.



Slika 8.1 Izrada animacije

Da bi se animacija spremila kao videozapis, potrebno je odabrati vrstu datoteke koju želimo spremiti, u ovom slučaju AVI .jpeg formatu, a na istom se mjestu može odabrati i lokacija spremanja datoteke, kao što je vidljivo na slici 8.2.



Slika 8.2 Odabir postavki za izradu animacije

9. REZULTATI

Za svaku od metoda izrađena je animacija. Svaka od animacija razlikuje se jedino u položaju i broju lišća koji su zasebno dodavani za svaku od metoda. Sve ostale postavke i okolina su jednaki za sve animacije.

Kako bi metode usporedili, usporedit ćemo izgled algi za odabrani frame te vidjeti postoji li razlika koja bi se vizualno mogla primijetiti.

Na slikama 9.1-9.5 prikazane su alge za *frame* = 40.

Slika 9.1 Frame 40 za odeint metodu

Slika 9.2 Frame 40 za Eulerovu metodu

Slika 9.3 Frame 40 za Heunovu metodu

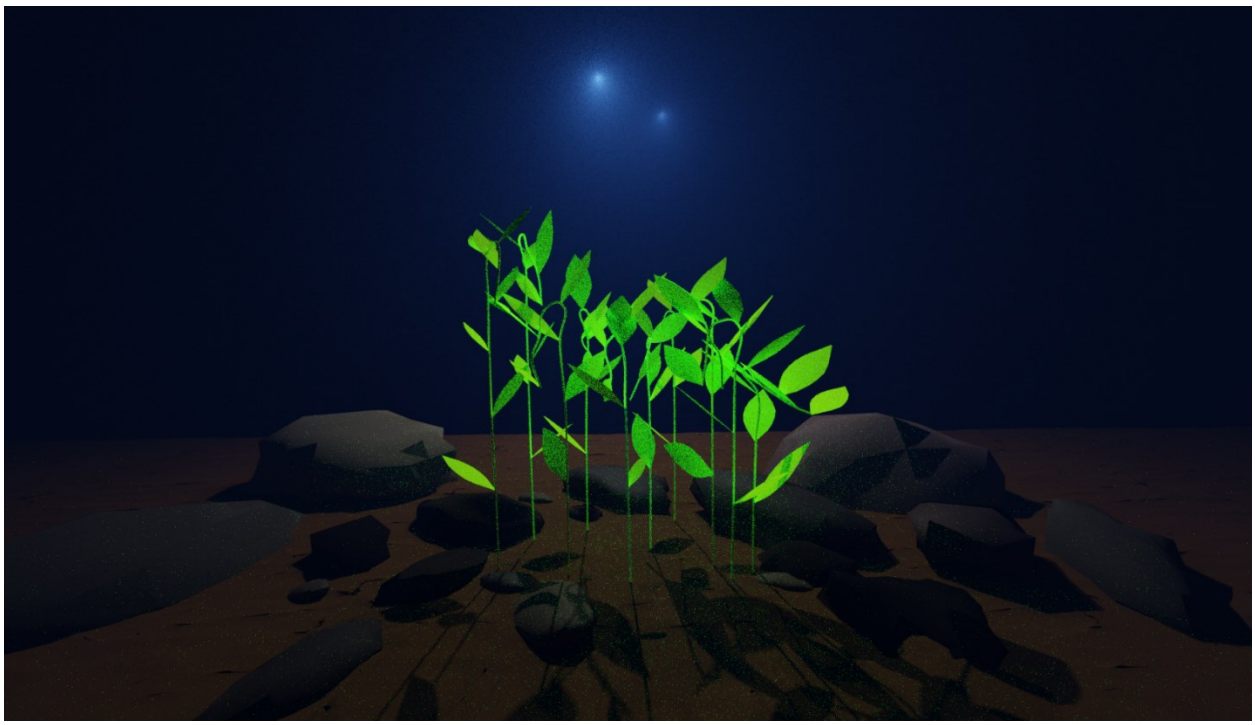
Slika 9.4 Frame 40 za RK4 metodu

Slika 9.5 Frame 40 za Midpoint metodu

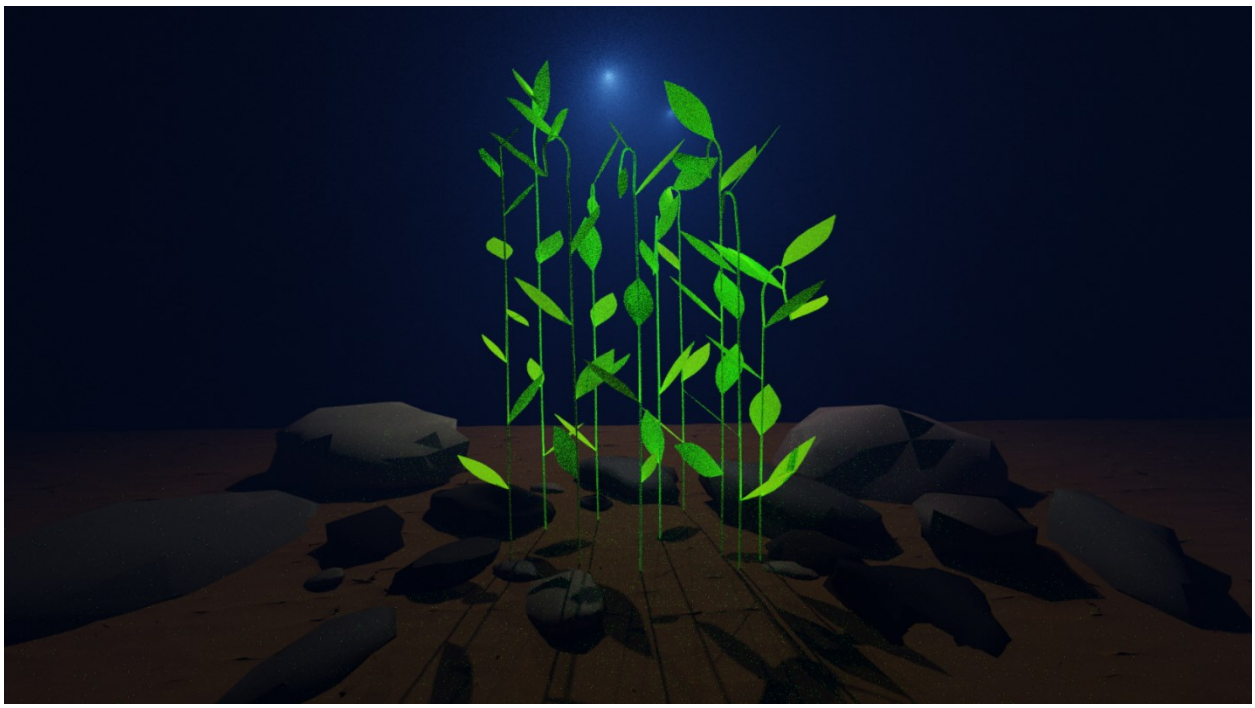
S obzirom na to da je teško vizualno pronaći razlike između navedenih metoda, zaključujemo da je naš slučaj bio jednostavan te da svaka od metoda daje jednako zadovoljavajuće rezultate.

9.1. Prikaz animacije u Cycles rendereru

Animaciju smo izradili i u Cycles rendereru koji daje vizualno bolje rezultate. Dok je za izradu animacije u Eevee rendereru računalu bilo potrebno nekih dvadesetak minuta, za izradu animacije u Cyclesu bila su potrebna preko tri sata. Također, izrada animacije u Cyclesu poprilično je zahtjevna za računalo, stoga možda nije najbolji izbor za izradu kratkih animacija kojima je cilj prikazati kako funkcionira određeni sustav. Prikaz izgleda scene vidljiv je na slici 9.6. i 9.7.



Slika 9.6 Animacija izrađena Cycles rendererom



Slika 9.7 Animacija izrađena Cycles rendererom

10.ZAKLJUČAK

Opisan je postupak izrade alge koristeći se modelom masa i opruga. Postavljene su sile koje djeluju na pojedinu masu i na pojedinu oprugu u sustavu, a sve se temelji na opisanom Newtonovom drugom zakonu. Opisane su različite metode rješavanja sustava običnih diferencijalnih jednadžbi te je napravljena usporedba rezultata dobivenih njihovim rješavanjem.

Za svaku metodu prikazana je implementacija numeričkog modela, kao i implementacija matematičkog modela.

Za svaku metodu izrađena je animacija uporabom *Eevee* renderera te se tako mogu vizualno usporeditit rezultati simulacije. Na prvi pogled sve metode daju jednaku animaciju alge, no u teoriji najbolja bi trebala biti metoda rješavanja putem Scipy modula, koja najbrže dolazi do rezultata.

Sa obzirom da je alga jedinka na koju utječu i vanjske sile, opisane su i postavljene jednadžbe za glavne vanjske sile koje djeluju u fluidu. Pozornost je obraćena i na materijal alge, pa su u jednadžbama uzete konstante koje odgovaraju stvarnom materijalu alge.

Prikazan je postupak modeliranja okoline te značajki alge u programu Blender. Opisan je način modeliranja lišća, podloge i kamenja koji sačinjavaju scenu te način kako dodati materijal svakom objektu.

Opisan je način izrade animacije i postavljanje kamere i svjetla koji animaciji daje realističnost i dinamičan izgled izmjenom svjetla i sjene. Predstavljene su dvije mogućnosti postavljanja kamere i svjetla, ručno ili preko koda.

LITERATURA

[1] S. Ivić, J. Škifić, S. Družeta, B. Crnković, M. Tuhtan, L. Grbčić, I. Lučin i M. Čavrak, u *Računarsko inženjerstvo uz programski jezik Python*, Rijeka, Tehnički fakultet Sveučilišta u Rijeci - Zavod za mehaniku fluida i računarsko inženjerstvo, 2019., p. 280.

[2] ScienceDirect, "Runge-Kutta Method," [Mrežno]. Dostupno na: https://www.sciencedirect.com/topics/mathematics/runge-kutta-method. [Pokušaj pristupa: 17. 07. 2022.].

[3] H. enciklopedija, "Coriolisova sila," [Mrežno]. Dostupno na: https://www.enciklopedija.hr/natuknica.aspx?ID=69182. [Pokušaj pristupa: 09 07 2022].

[4] Britannica, "Causes of ocean currents," [Mrežno]. [Pokušaj pristupa: 09. 07. 2022].

[5] L. Kranjčević, "Mehanika fluida - Skripta za studente Tehničkog fakulteta Sveučilišta u Rijeci," 2019.

[6] B. Gaylord, "Modulation of wave forces on kelp canopies by alongshore currents," 2013.

[7] Blender.org, "About us," [Mrežno]. Dostupno na: https://www.blender.org/about/. [Pokušaj pristupa: 12. 07. 2022.].

[8] Blender, "Introduction to Nodes," [Mrežno]. Dostupno na: https://docs.blender.org/manual/en/2.79/render/blender_render/materials/nodes/introduction.html. [Pokušaj pristupa: 07. 2022.].

[9] US3DPrinters, "What is rendering in Blender? Simply explained!," [Mrežno]. Dostupno na: https://www.us3dprinters.com/rendering-in-blender/. [Pokušaj pristupa: 01. 08. 2022.].

[10] Blender, "Rendering," [Mrežno]. Dostupno na: https://www.blender.org/features/rendering/. [Pokušaj pristupa: 25. 07. 2022.].

[11] Blender, “GPU Rendering,” [Mrežno]. Dostupno na: https://docs.blender.org/manual/en/latest/render/cycles/gpu_rendering.html. [Pokušaj pristupa: 07. 08. 2022.].

SAŽETAK

U ovom radu predstavljena je problematika izrade simulacije alge primjenom metode masa i opruga. Opisane su sile koje djeluju unutar samog sustava kao i vanjske sile koje djeluju na algu. Dane su usporedbe metoda kojima se sustav jednadžbi može riješiti te je izrađena animacija kao rezultat simulacije.

Opisana je implementacija numeričkog modela koji podrazumijeva rješavanje sustava jednadžbi danih u matematičkom modelu.

Rad se nadalje bavi modeliranjem scene i lišća alge u programu Blender, gdje se opisuju postupci i alati korišteni za izradu realistične scene. Opisuje se postavljanje kamere i svjetla. Daje se kratki pregled renderera koje Blender nudi kao i njihove glavne značajke. Kao završni korak prikazuje se postupak izrade same animacije.

Ključne riječi: model masa i opruga, Blender, animacija, alga.

SUMMARY

In this paper we present the issue of making a simulation of algae, using the mass spring method. The forces acting within the system as well as the external forces acting on the alga are described. Comparisons of the methods by which the system of equations can be solved are given, and an animation is made as a result of the simulation.

The implementation of the numerical model is described, which involves solving the system of equations given in the mathematical model.

The paper also deals with the modeling of the scene and algae leaves in Blender, where the procedures and tools used to create a realistic scene are described. Camera and light setup is described. A brief overview of the renderers offered by Blender is given as well as their main features. As a final step, the process of creating the animation itself is shown.

Key words: mass and spring model, Blender, animation, algae.