

Zadanie projektowe 2

Grafika 3D

Volodymyr Tsukanov

Etap 1. Budowa sceny

Treść zadania

Należy zimportować i ułożyć w blenderze wszystkie elementy sceny. Nasz obiekt centralny może być np. ustawiony na stoliku w pokoju albo stać przy ognisku albo... (Elementy wystroju można stworzyć samodzielnie lub pobrać z witryn udostępniających modele na otwartych licencjach – w takim przypadku należy dołączyć plik tekstowy wyszczególniający: co, skąd i na jakiej licencji zostało umieszczone na scenie). Dla zbudowanej sceny należy ustawić odpowiednie oświetlenie – oświetlające całą scenę i eksponujące obiekt główny.

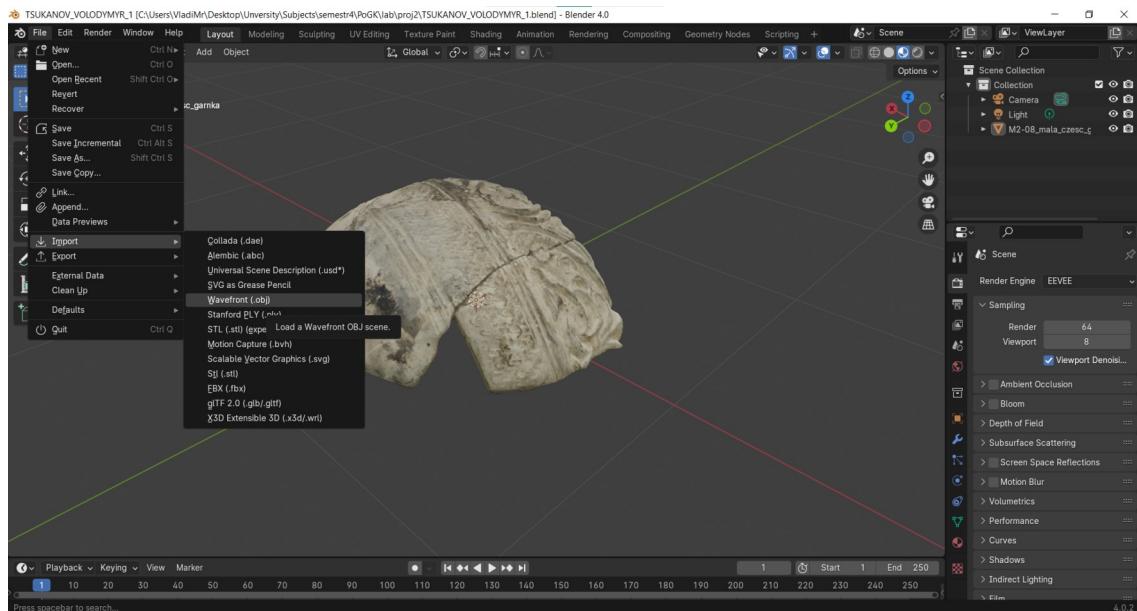
Ustawić kamerę tak aby obiekt był dobrze widoczny a całość wyglądała ciekawie i wykonać rendering (minimalna rozdzielcość 1920x1080).

Pliki wynikowe: [TSUKANOV_VOLOODYMYR_1.blend, etap1_e-32sp.png](#)

Wykonanie

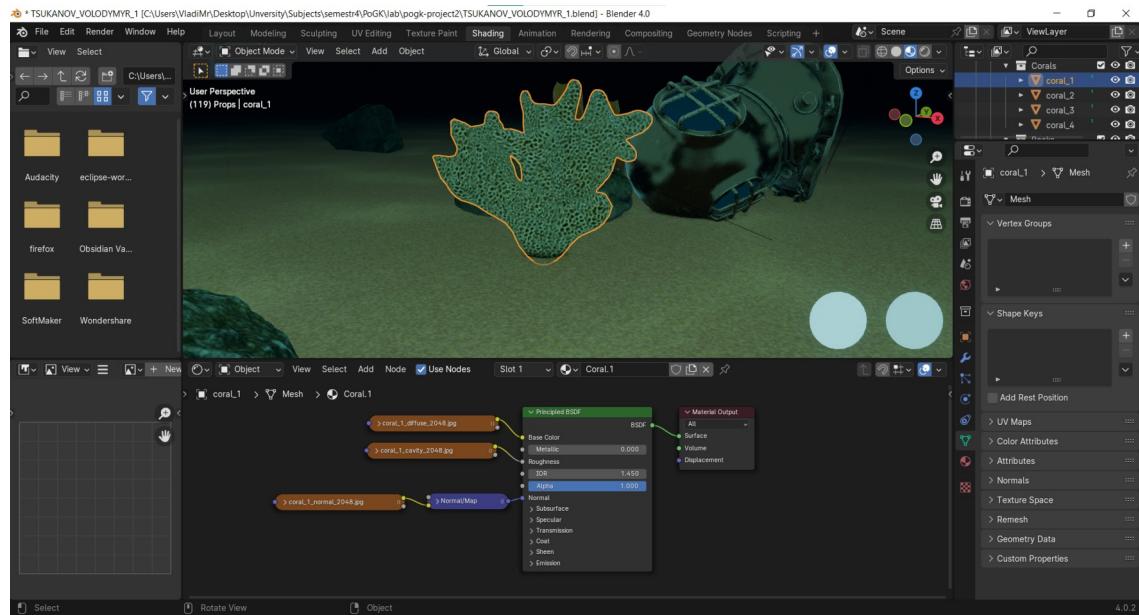
1. Importowanie

Menu File > Import > format importowanego pliku

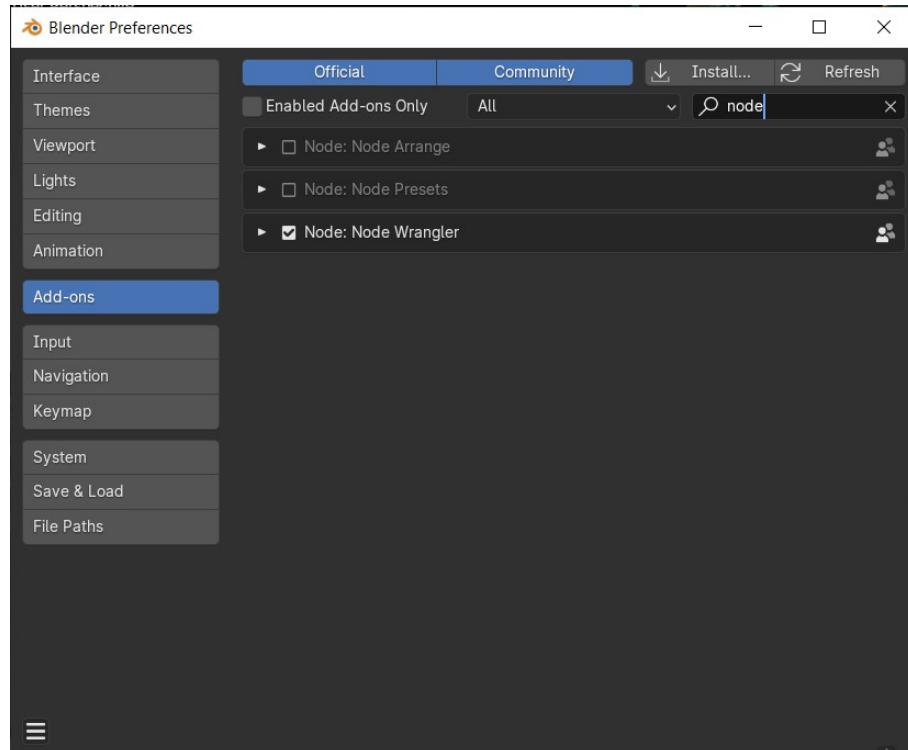


Natychmiast należy naprawić materiał importowanego obiektu:

Wybieramy importowany obiekt. **Widok Shading > Shader Editor** (okno na dole) naprawiamy lub dodajemy tekstury do odpowiednich wejść shadera.

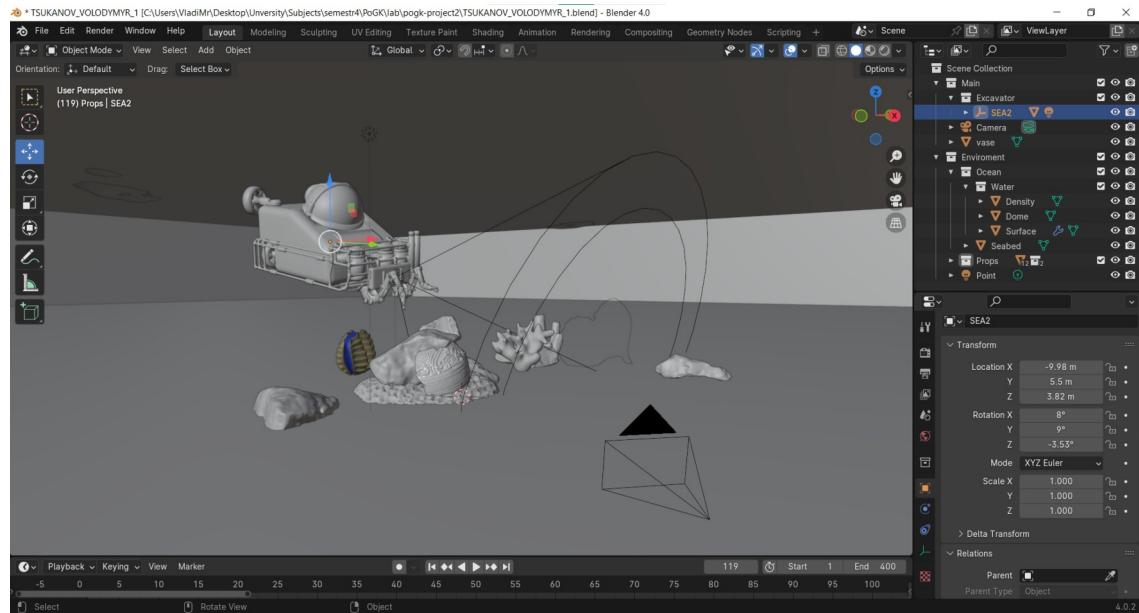


Dla przyspieszenia tego procesu można skorzystać z *Node Wrangler* (Włączamy go w **Menu Edit > Preferences** i, wybieramy materiał klikamy na *shader*, **Ctrl+Shift+T**, wybieramy tekstury)

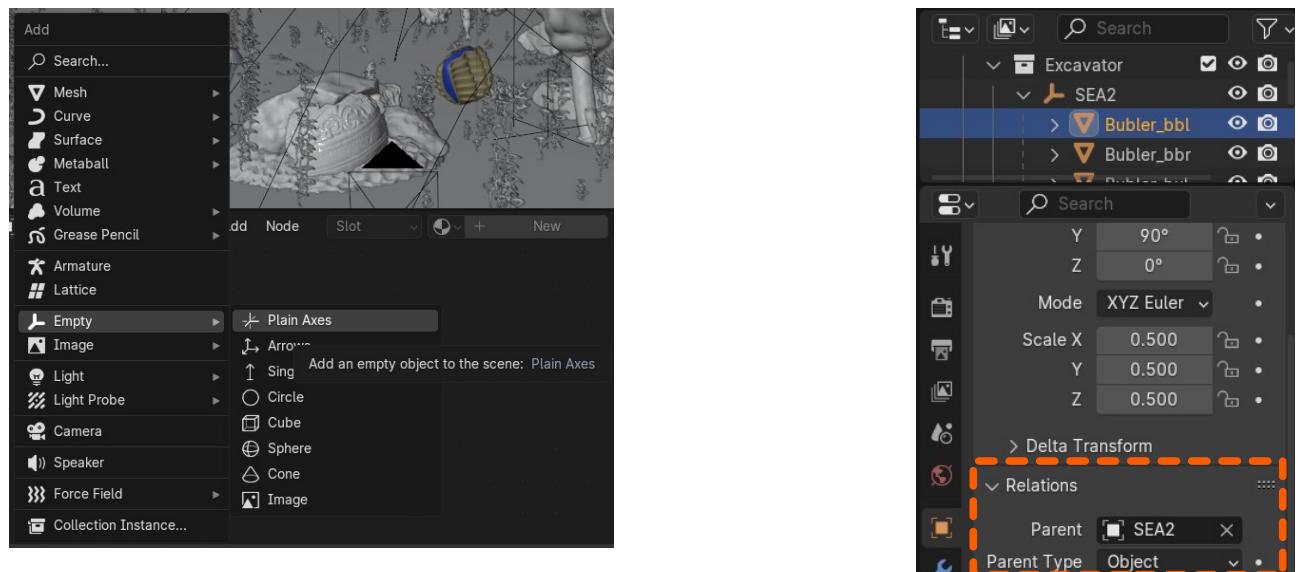


2. Grupowanie i rozmieszczenie

Widok Layout. Korzystając z narzędzi **Move**, **Rotate**, **Scale** oraz grupując obiekty w kolekcji odzyskujemy następującą scenę.



Przydatny trik do grupowania kilku obiektów (parenting **Ctrl+P**): **Shift + A > Empty** tworzymy nowy pusty obiekt. Dalej we wszystkich preferowanych obiektach w zakładce **Relations** wybieramy niedawno utworzony pusty obiekt jako **Parent**.



3. Środowisko

Ponieważ obiekt główny będzie znajdował się pod wodą należało by symulować oświetlenie i ogólny widok sceny używając odpowiednie efekty:

- **Density**

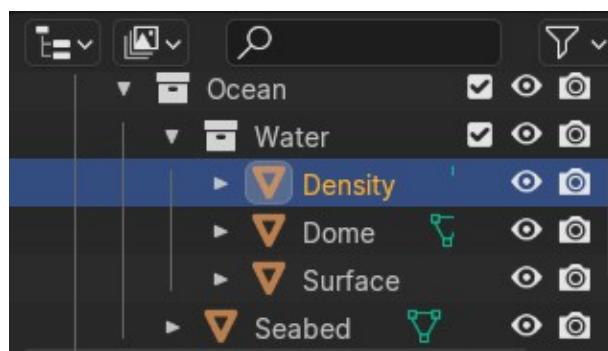
Symulacja gęstości wody zrobiona za pomocą *shadera Principled Volume*.

- **Dome**

Chroni obiekty w środku przed światłem (pod wodą światło nie może świecić z dołu lub ze stron).

- **Surface**

Powierzchnia wody – **Ocean modifier**, **Refraction shader**.



4. Rendering

Na tym etapie użyłbym **Eevee** render engine. W *oknie Properties* (ustawienia; po prawej stronie) > **Render Properties** należy włączyć **Bloom**, **Ambient Occlusion**, **Screen Space Reflections** oraz inne preferowane efekty. Ale trzeba pamiętać o wydajności, na tym etapie byłoby zbędnym używać wysokie ustawienia renderingu.

Jak wszystko ustawione, wciskamy **F12** (lub *Menu Render > Render Image*) i czekamy (tego razu nie długo).

Etap 2. Korekta jakości modelu obiektu

Treść zadania

Etap ten przewiduje dokonanie próby poprawy efektu wizualizacyjnego modelu głównego. Możliwe działania to np.:

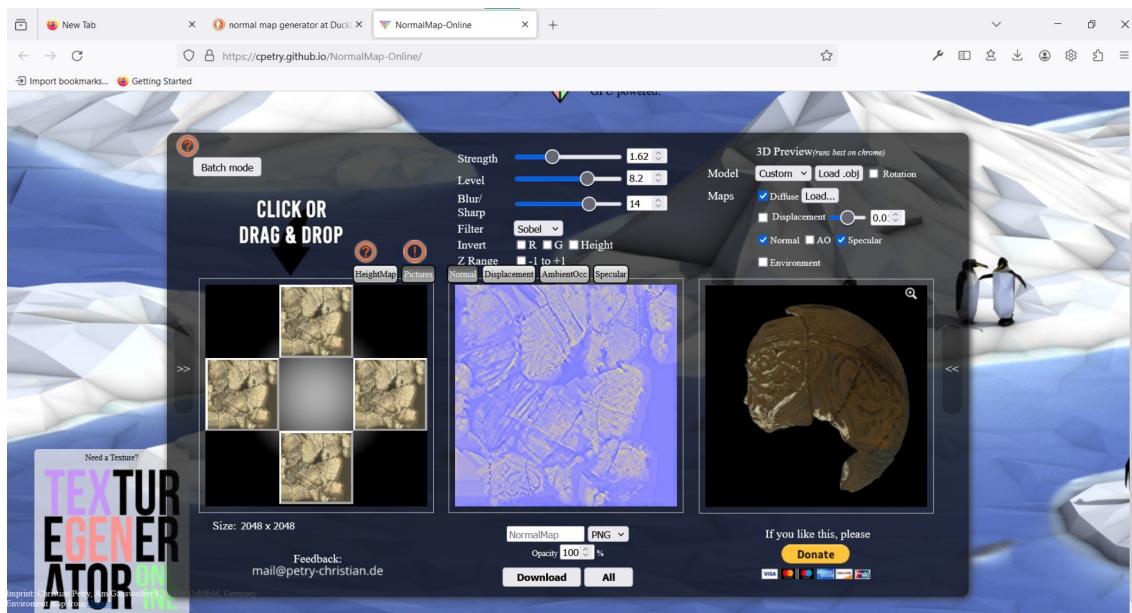
- Działania na teksturowe – globalne poprawy kolorów, zwiększenie rozdzielczości, poprawy braku ostrości, eliminacja zakłóceń (np. napisy)
- Wytworzenie tekstury normal map (wykorzystując aplikacje zewnętrzne np. Awesome Bump, Shader Map) i użycie jej przy renderingu.
- Wytworzenie tekstur odpowiadających za połysk (Roughness, metalness...) i użycie ich w renderingu.
- Modyfikacje parametrów materiału obiektu w blenderze by uzyskać lepszy efekt renderowania.
- Dodanie obrazu tła do sceny (wyszukanie odpowiedniego tła dla zbudowanej sceny i takie ustawienie sceny i kamery by scena dobrze się komponowała w obrazie)

Pliki wynikowe: [TSUKANOV VOLODYMYR 2.blend](#), [etap2_c-1008_d.png](#), [etap2_c-1008_p.png](#)

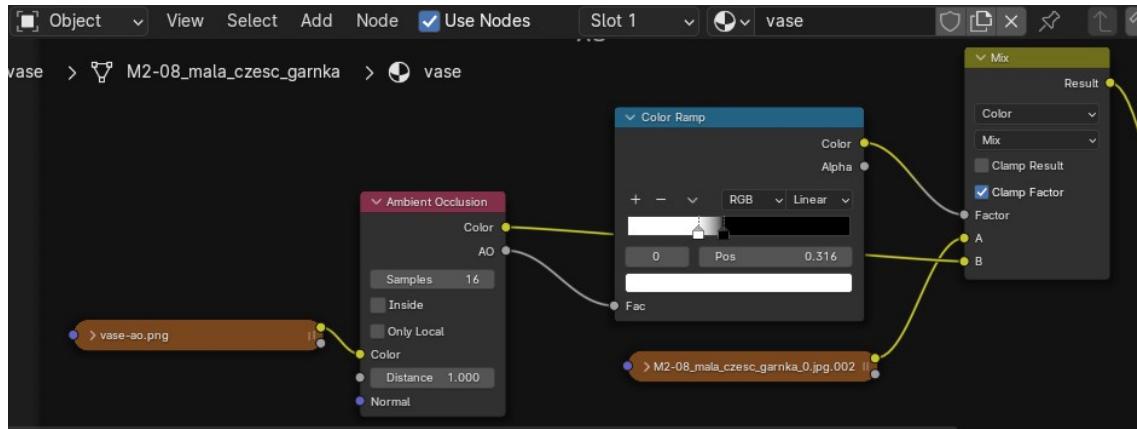
Wykonanie

1. Tekstury dla garnka

Korzystając z [narzędzia do generowania tekstur](#) wygenerowano **Normal**, **Specular**, **AmbientOcclusion** dla garnka.

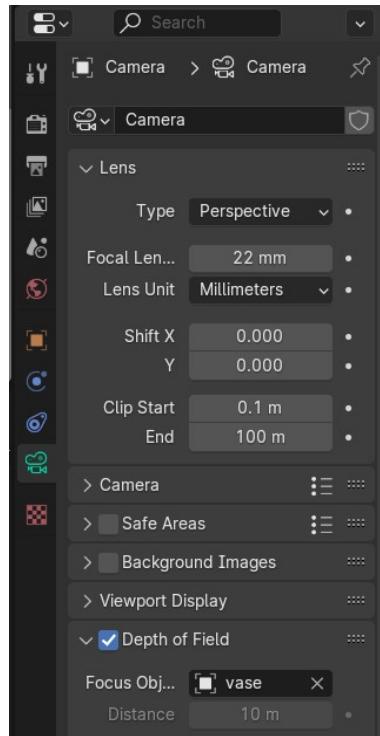


Dodanie tekstur do materiału obiektu głównego. Poniżej podany sposób dołączenia AO i albedo do **Base Color**. Także podobnie do [etapu 1](#) zostały dodane **Normal** i **Specular**.



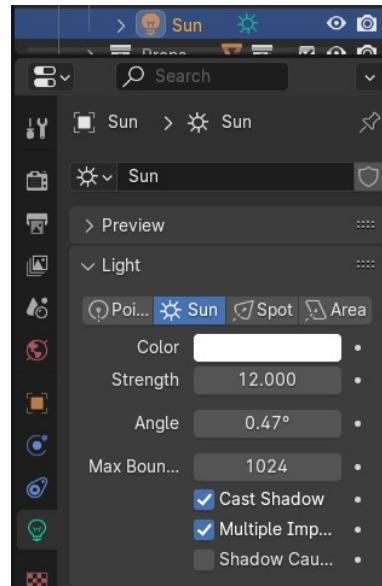
2. Kamera

Zmniejszamy parametr **Focal Length** do 22mm dlatego że scena jest pod wodą i dla lepszej widoczności wszystkich obiektów będziemy umieszczać ich blisko kamery. Dodatkowo ustawiamy **Depth of Field** na garnek dla symulowania działania rzeczywistej kamery.



3. Oświetlenie globalne

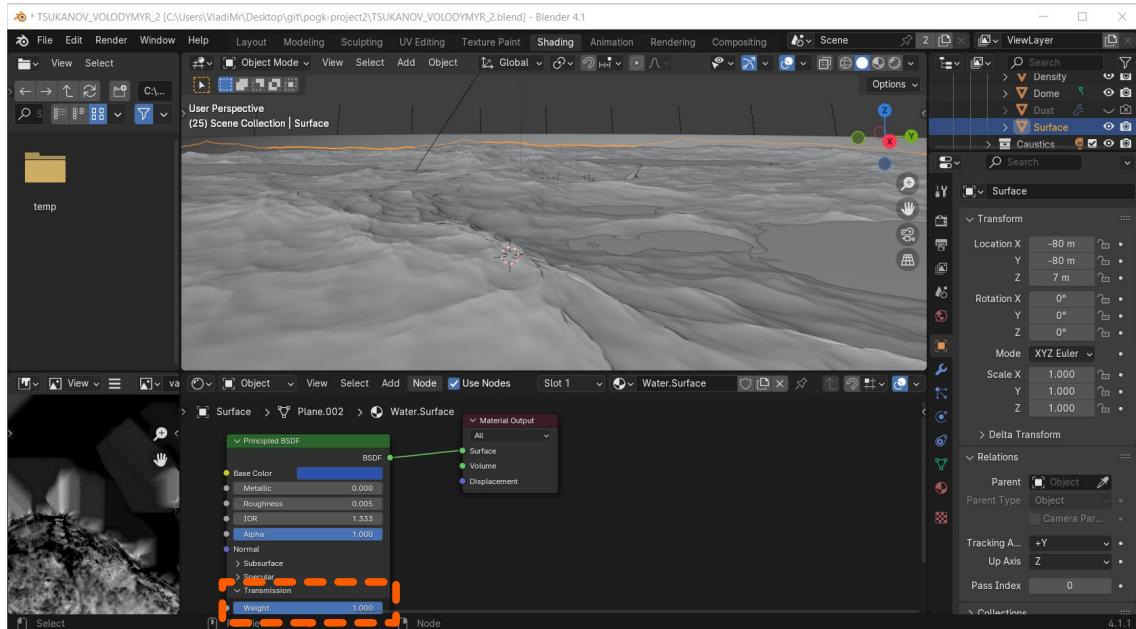
Tworzymy nowe źródło światła w scenie. Uznałem za pasujące następujące ustawienia tego światła.



4. Materiały

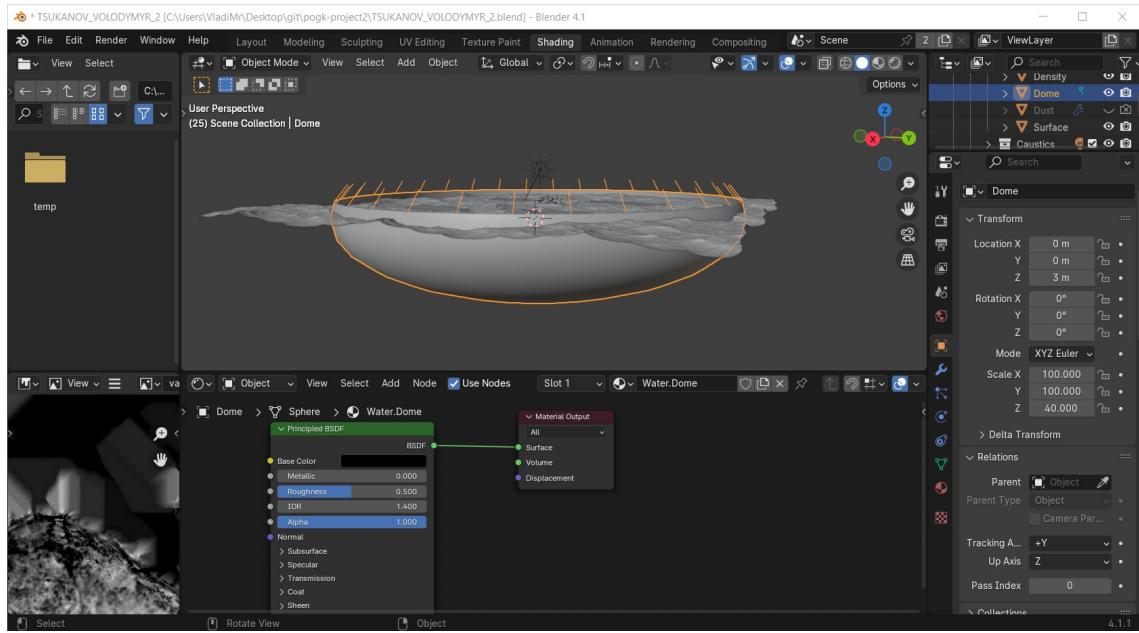
- **Surface**

Jako *shader* dla powierzchni morza wybrany został prosty **Principled BSDF** z **Roughness** bliskim 0, **IOR** równym wartości wody (1.333) oraz maksymalnej transmisji (**Transmission**), dla lepszego przepuszczania światła przez swoją powierzchnię. Jeśli czytasz, to gratulacje bo dowiedziałeś się że trzeba dodatkowo wyłączyć opcję shadow w preferencjach obiektu powierzchni.



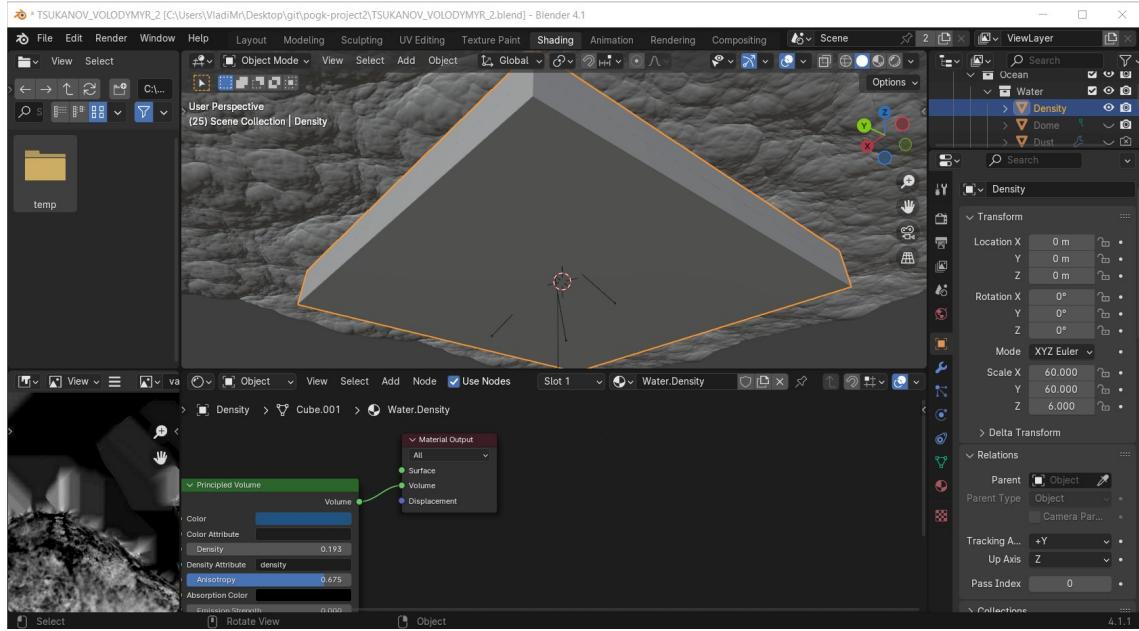
• Dome

Czarna kula chroniąca przed zaświeceniem z boków.



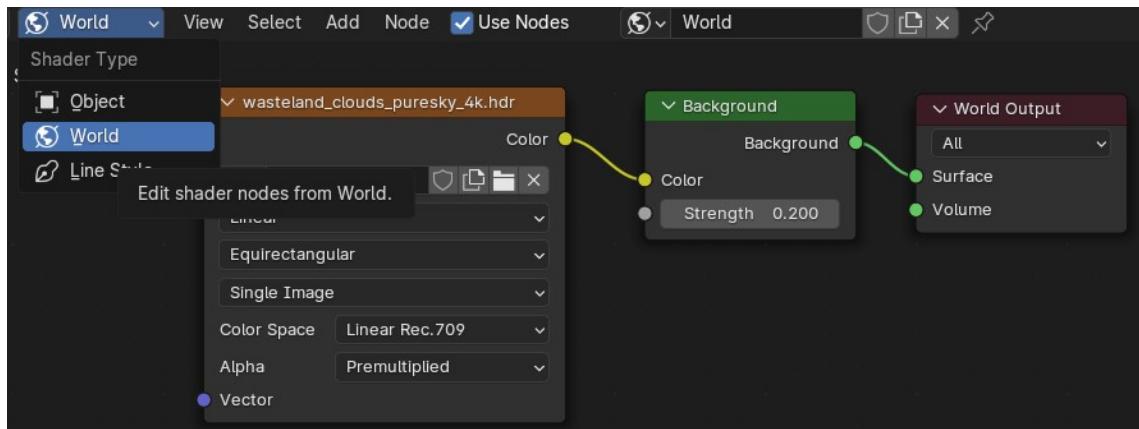
• Density

Woda... lub chociaż to co ją przypomina. **Principled Volume shader** idący do **Volume** (nie do **Surface**). Co do ustawień: im większe **Density**, tym słabiej widać w morze (gęstsza woda); **Antisotropy** definiuje jak gęstość rozprasza po bokach obiektu, im większa ta wartość, tym bardziej przezroczyste boki wody.



- **World shader**

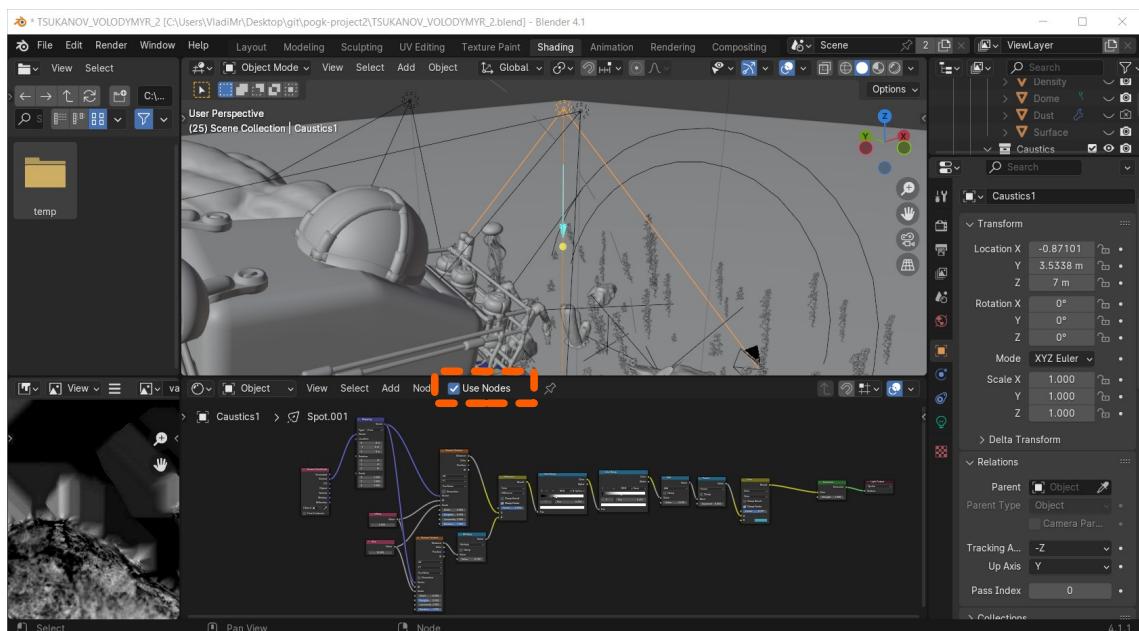
Wykorzystana została 4K **HDRI** tekstura nieba, dla odzwierciedlenia go na wodzie. Jak w rzeczywistości kiedy jesteśmy pod wodą i patrzymy w górę, widać niebo. (Dla ułatwienia znalezienia odpowiedniego *noda*, szukamy **Environmental Texture**).



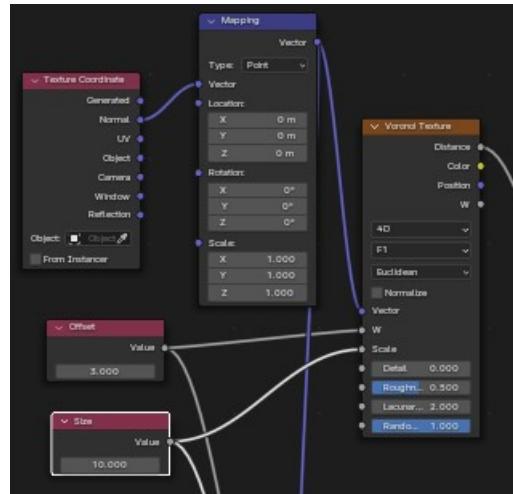
- **Caustics**

Pomimo nieba pod wodą są widoczne promienie słońca. Jest to jedna z najfajniejszych i przepięknych rzeczy moim zdaniem. Chociaż mój laptop tak bym nie myślał, dlatego że symulowanie tego efektu w grafice jest jednym z najtrudniejszych zadań. Zasymulować można wszystko, zależy jak długo będziemy czekali przy renderowaniu. Więc upraszczamy to zadanie na ile się da. (Nie jestem wynalazcą tej metody zostawiam link do wideo).

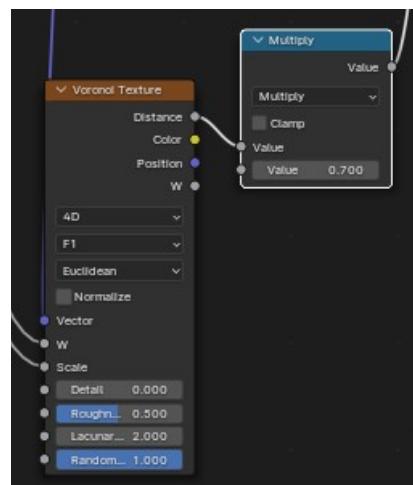
Materiał był zrobiony za pomocą odejmowania 2 tekstur **Voronoi Noise**, ich **Color Ramp**-owania oraz kilku innych operacji matematycznych (dodawanie i stropianie) dla utworzenie źródła światła **Caustics**. Ważne włączyć opcję **Use Nodes** przed konstruowaniem *shadera*.



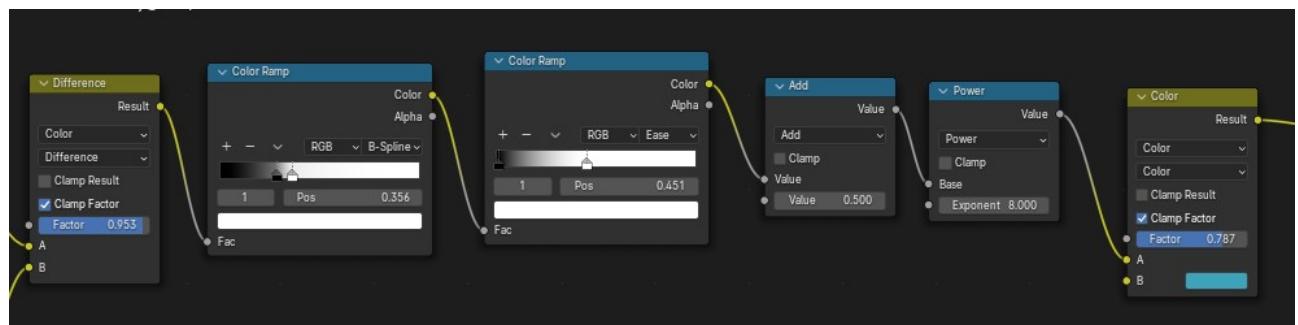
Dodanie **Vector**-a pozycji na początku shadera, wykorzystanie custom inputs dla wejścia W oraz Scale (synchronizacja tekstur, potrzebne dla animacji na etapie 3).



Mnożymy drugą teksturę przez wartość bliską 1, żeby pozbyć identyczności.



Odejmujemy wyjścia tekstur i modyfikujemy (na żywo patrząc na okno **Viewport**).



Zmodyfikowany sygnał do noda **Emission** i na wyjście *materialu*.



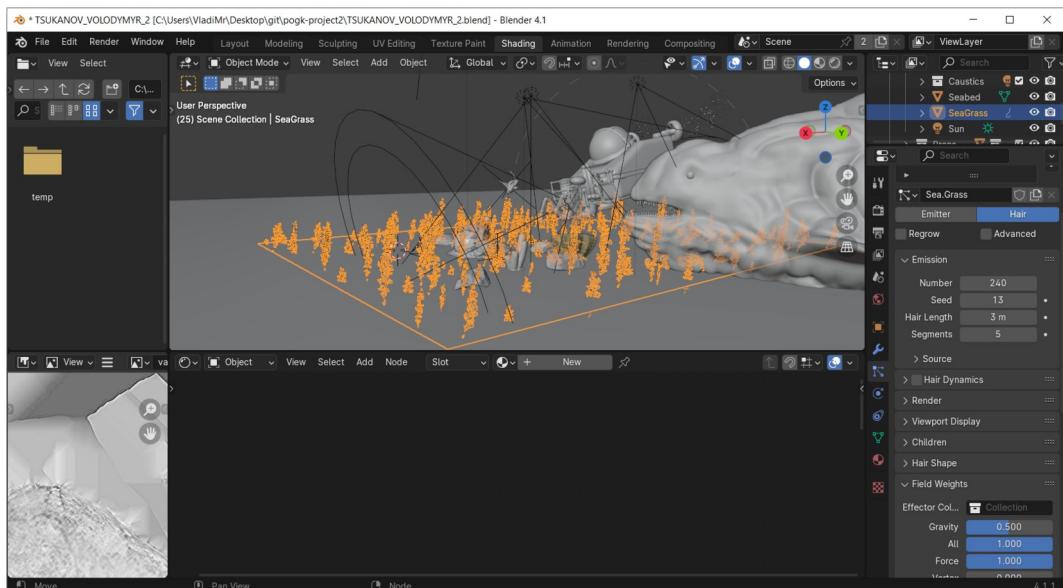
5. Particles

Na scenie tworzymy niewidoczną dla rendera *kolekcję* gdzie zamieścimy wszystkie obiekty cząstek.



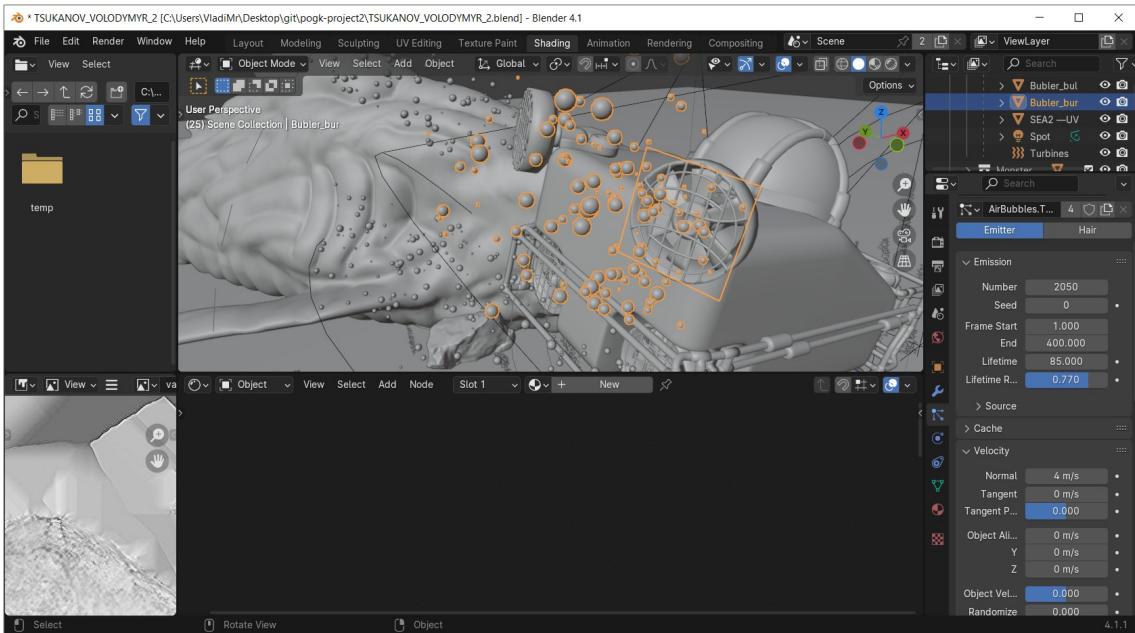
- **Sea Grass**

Żeby na scenie nie było pustych miejsc oraz dla realizmu dodajemy trawę. Tworzymy **Plane** (**Shift+A**) i dodajemy **Particle System** do niego. **Type = Hair, Render As = Collection → SeaGrassGroup**.

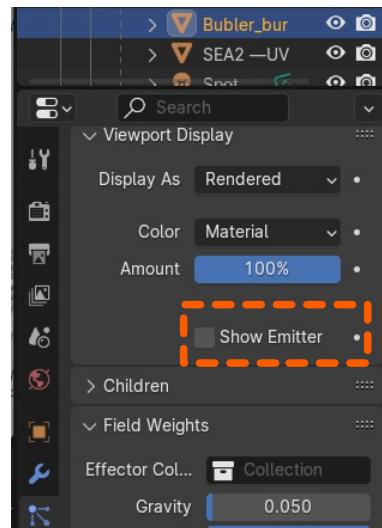


- **AirBubbles**

Na renderze tego nie widać ale oni są. Tym bardziej ten system częstek przyda się na 3 etapie.

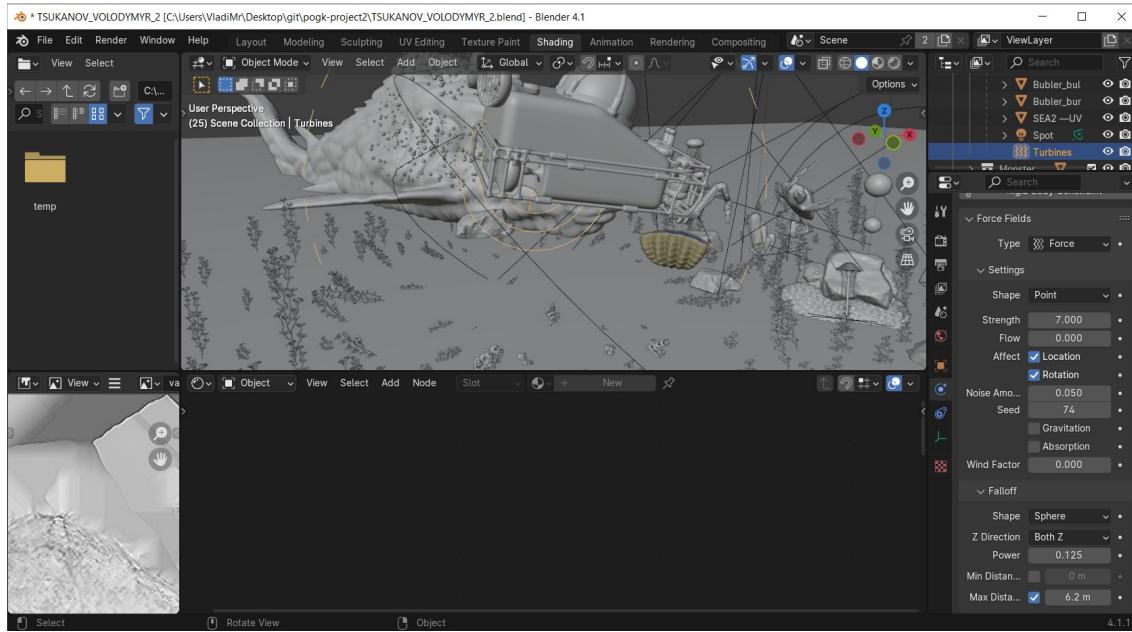


Dopóki pamięć nie zawiodła podam ważną rzecz, należy wyłączyć opcję **Show Emitter**, bo na renderze pojawi brzydkie białe **plane** i będzie dobrze jeżeli zostanie zauważony na pierwszych *samplerach*, a nie pod koniec renderingu > ___ <



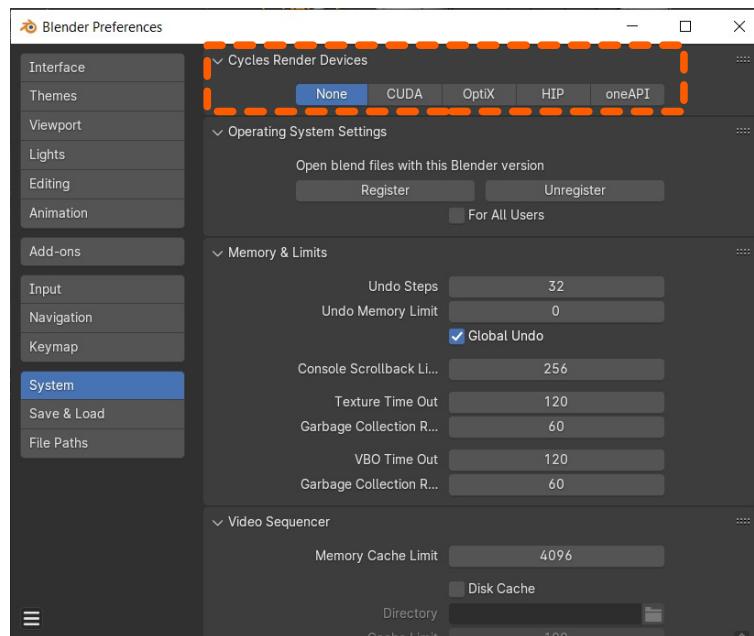
6. Forces

Dlaczego? Bo umiem (¬‿¬) ↗

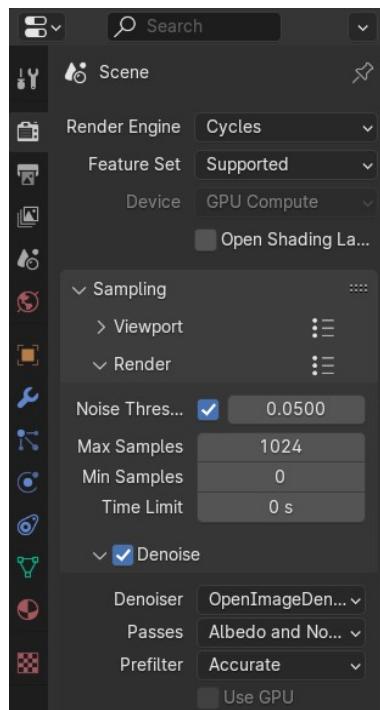


7. Render

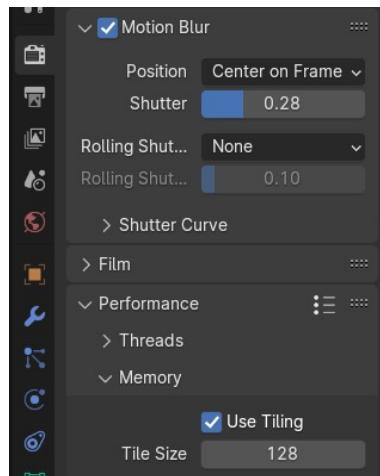
Są mnóstwo metod katowania, jedną z nich jest renderowanie grafiki 3D. Więc należy jak-najbardziej zmniejszyć czas tego procesu. *GPU* pomoże w tym (jeśli sprzęt posiada taki, w przeciwnym wypadku należy znaleźć ten co posiada (¬‿¬), **Ctrl+, > System > Render Devices** wybieramy pasującą opcję (**CUDA – Nvidia, OptiX – Nvidia RTX, HIP – AMD, oneAPI – Intel**).



W ustawieniach **Render**



Włączamy **Motion Blur** i jeśli nie ma *GPU* lub mało VRAM włączamy **Use Tiling** (dla *CPU* małe wartości, dla *GPU* – jak-najwięcej ile da).



Etap 3. Animacja

Treść zadania

Mając gotową scenę wykonaj krótką animację.

Pliki wynikowe: [TSUKANOV VOLODYMYR 3.blend](#), [etap3_c.mp4](#)

Wykonanie

1. Podział na podetapy

Najważniejszym z wszystkich etapów każdego filmiku, niezależnie od długości, jest przemyślenie scenariusza oraz rozbicie chronologii na podetapy. I to działa nie tylko w filmikach. Jak pojawi się idea trzeba zapisać ją i dobrze opracować, kiedy początkowy zmysł obrośnie dodatkowymi szczegółami można dzielić go na podetapy, a podetapy na małe historyjki dla obiektów. Pełny opis w pliku [README.md](#).

- ***Above the Ocean***

Aktory: statek towarowy, deszcz, błyskawica

Klatki: 1 – 87

- ***Near the Surface***

Aktory: żółw, meduza, ryby1

Klatki: 88 – 314

- ***The Sinking Ship***

Aktory: statek towarowy, ryby2, meduza2, garnek, zegarek kieszonkowy

Klatki: 315-499

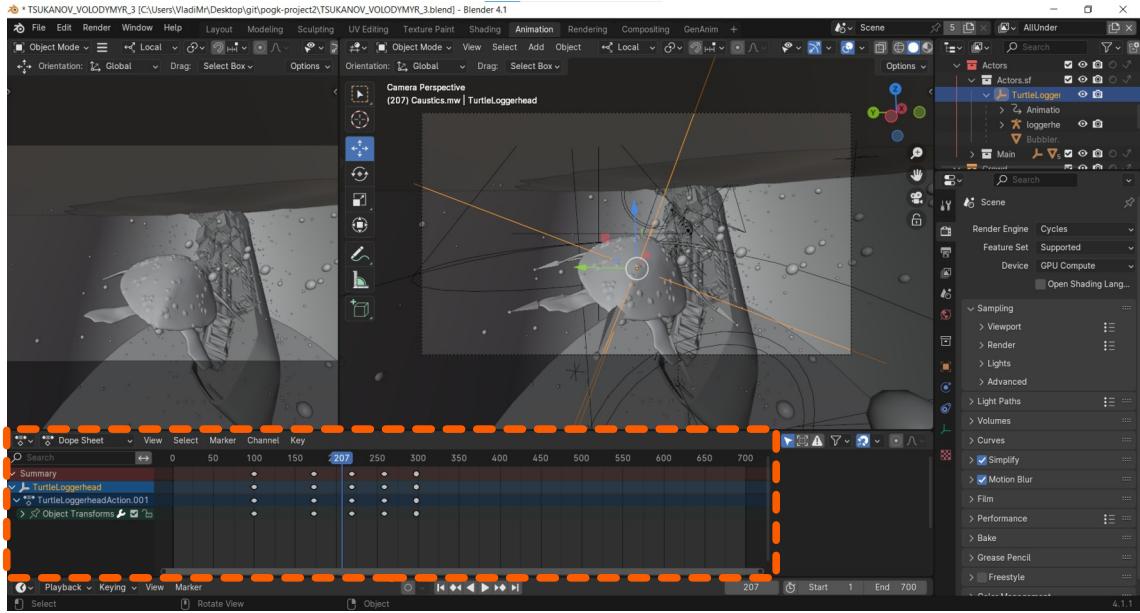
- ***Backwards***

Aktory: garnek, zegarek kieszonkowy, statek towarowy

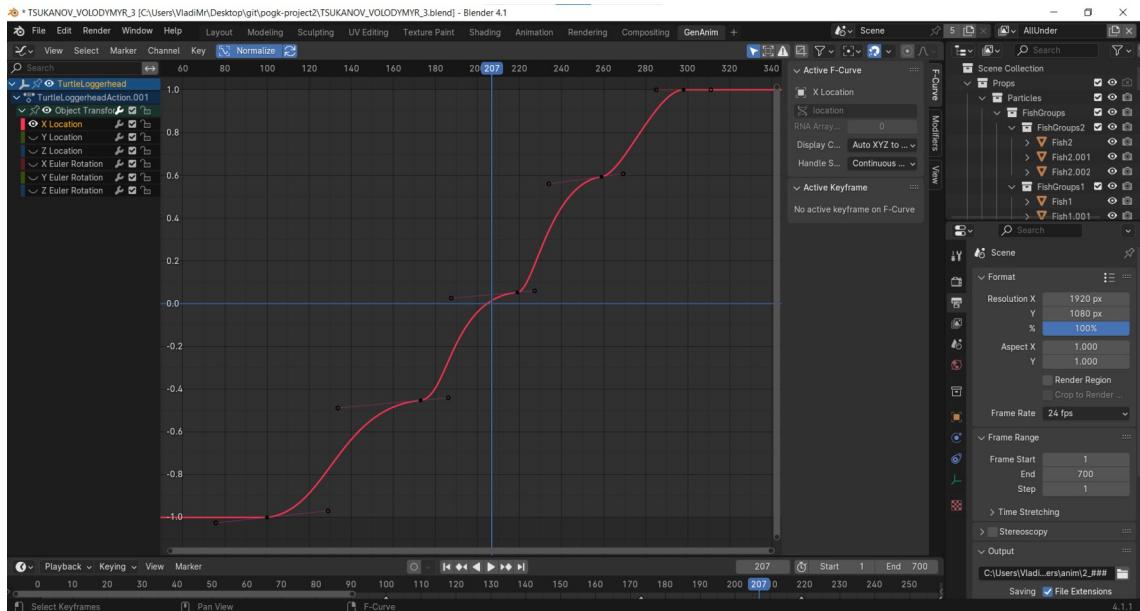
Klatki: 500-700

2. Animacja

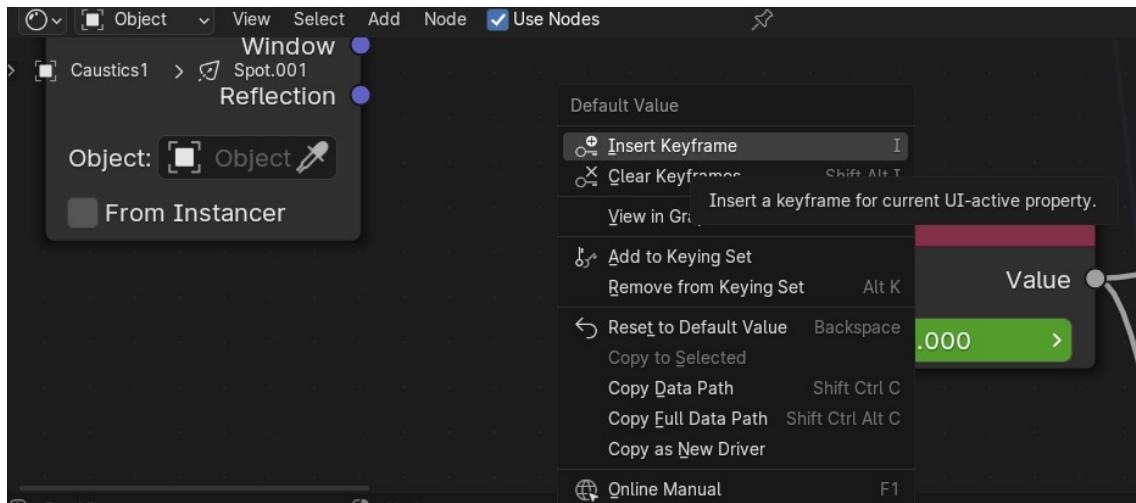
Przełączając na widok **Animation** animujemy poszczególne obiekty (np. żółw), przemieszczając pomiędzy *frejmany* ustawiamy *keyframe'y* (zaznaczamy pole, które chcemy zanimować wciskamy **I** dla utworzenia nowego *keyfram-a* **Alt+I** dla zamiany istniejącego).



Także dla ulepszenia rezultatu trzeba pozmieniać przyspieszenie pomiędzy punktami animacji (np. stosować **easy-in**, **easy-out**).

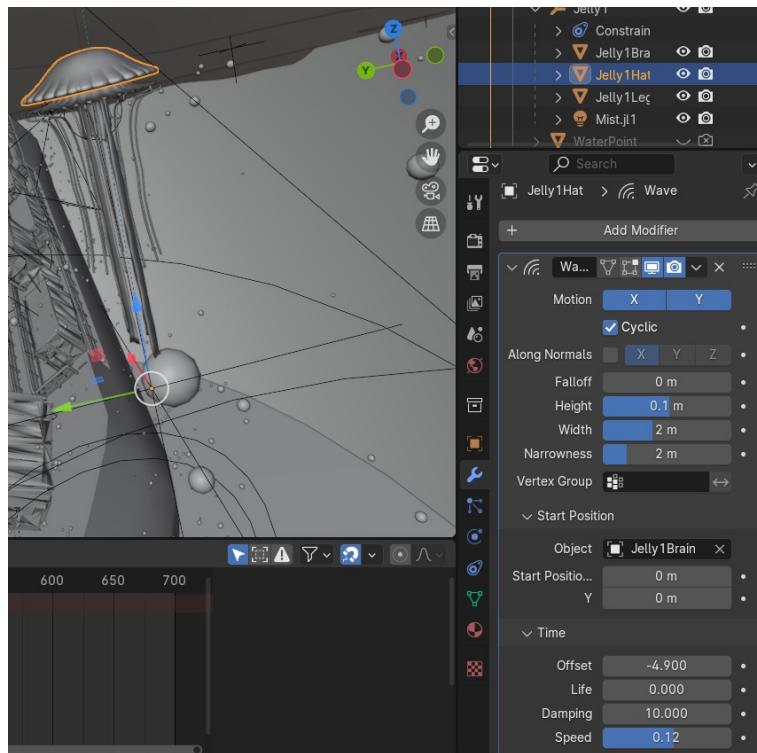


Animowania wartości źródła światła **Caustics** (zgodnie z [punktem 2.4](#)).

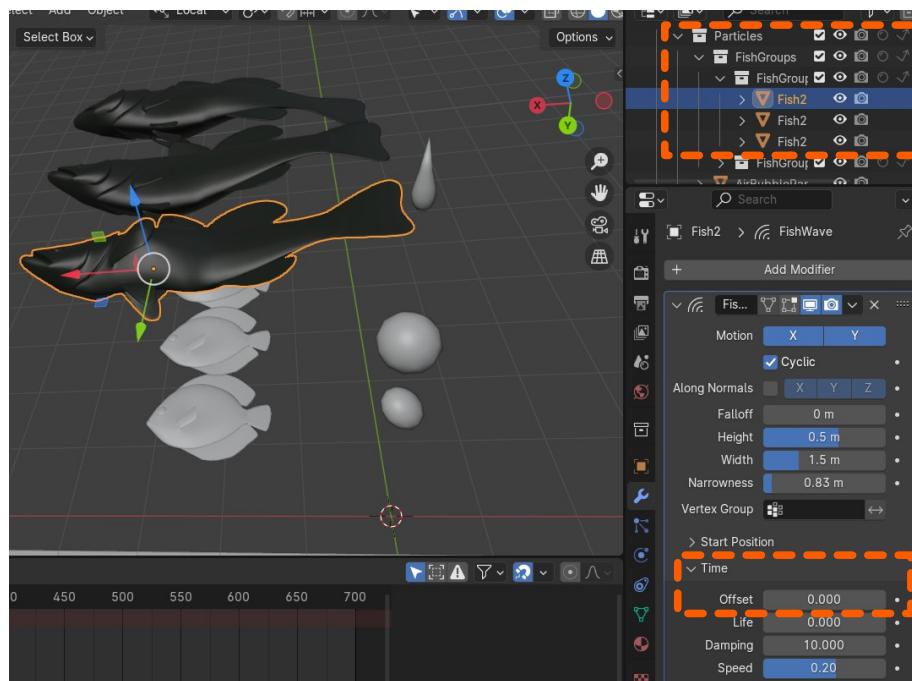


3. Modyfikatory

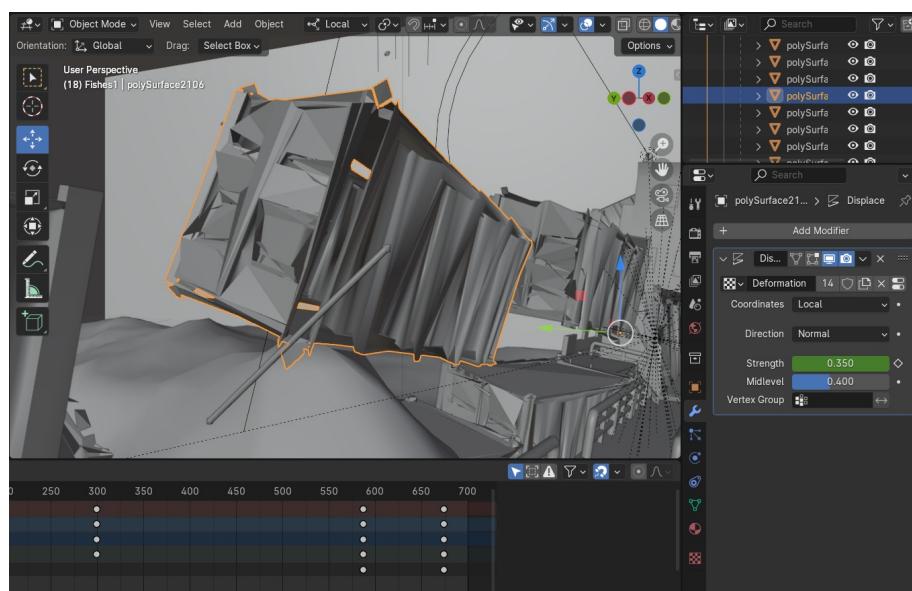
Dla oszczędzania czasu obiekty drugiego planu można animować bez użycia *animatora*, zamieniamy go na *modyfikator*. Dla ryb oraz meduz wykorzystamy **Wave modifier**.



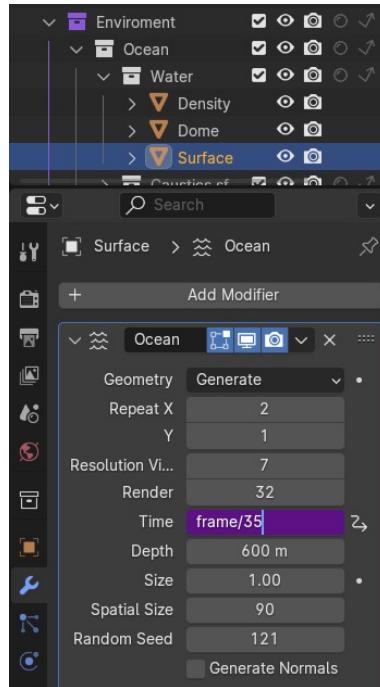
Nie zapominamy o wcześniej utworzonej *kolekcji* dla częstek. Na scenie będą 2 grupy ryb, dlatego tworzymy 2 **kolekcje**, po 3 ryby w każdej (im więcej ryb, tym mniejsza monotonia grupy) i dla każdej nowej ryby robimy inny **Offset**. Ważne, duplikować powtarzające obiekty z użyciem **Alt+D** (nie **Shift+D**), zatem my oszczędzamy resursy komputera (nie tworzymy pełną kopię z oddzielną siatką *mesh*, a korzystamy z tej samej siatki), tak możemy mieć milion kopii tych ryb ale korzystających z tego samego *mesh*-a.



Dla deformacji kontenerów wykorzystano *modyfikator Deformation*. W tym przypadku bez animowania nie wyjdzie, animujemy parametr **Strength**.



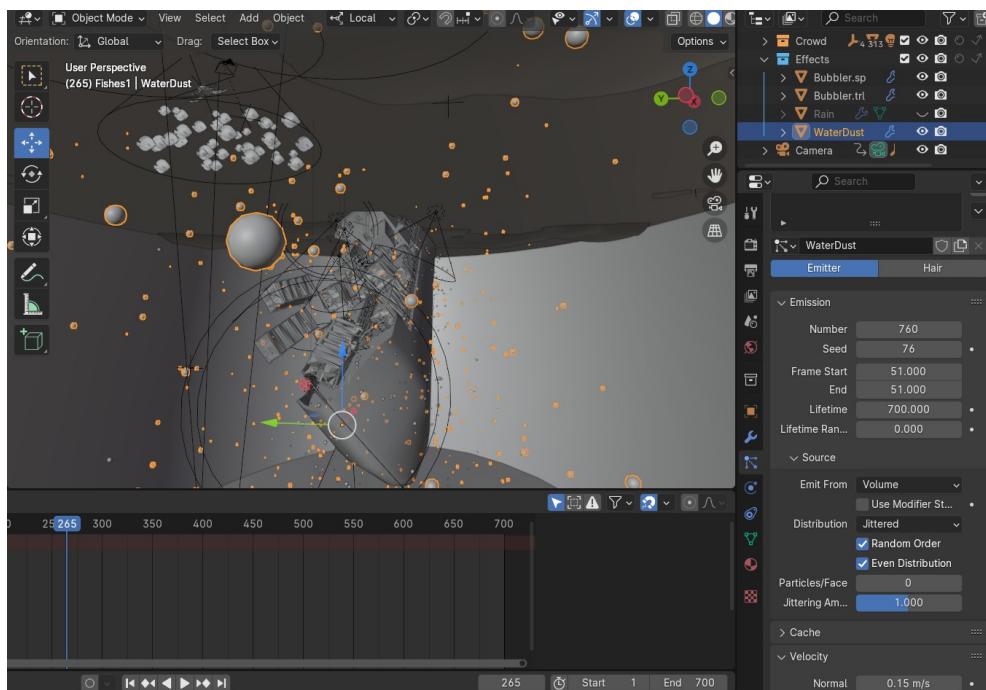
Morze animujemy za pomocą *drivera* (dzieli indeks *frame'a* na określoną stałą).



4. Particles

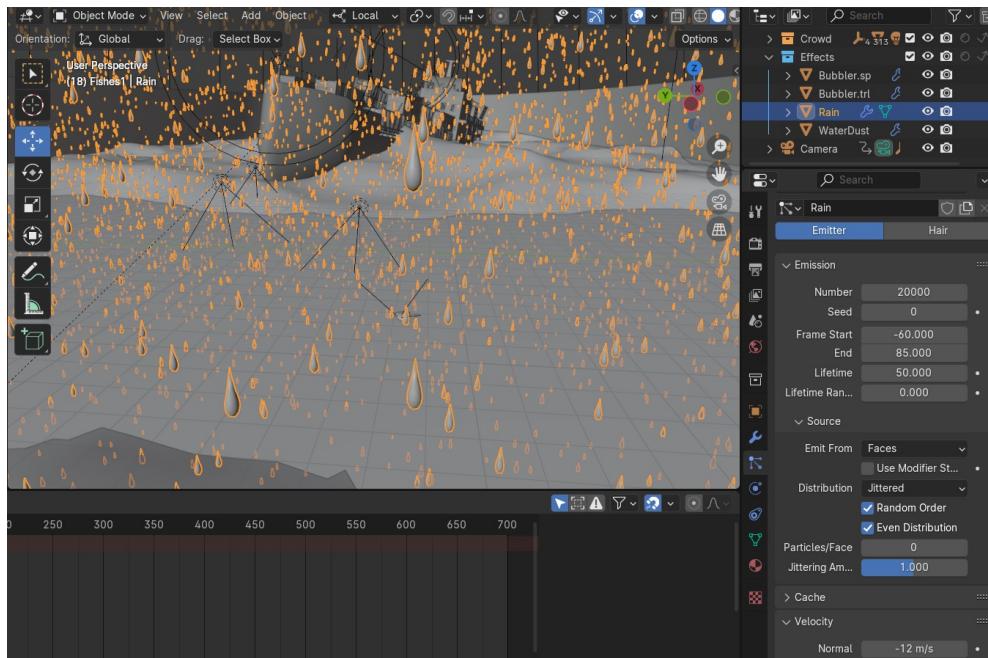
- **WaterDust**

Morze nie bez śmieci. Niestety, wśród nich, pomimo odpadów organicznych, można znaleźć ślady działalności człowieka ↗ ↘ ↙ ↖.



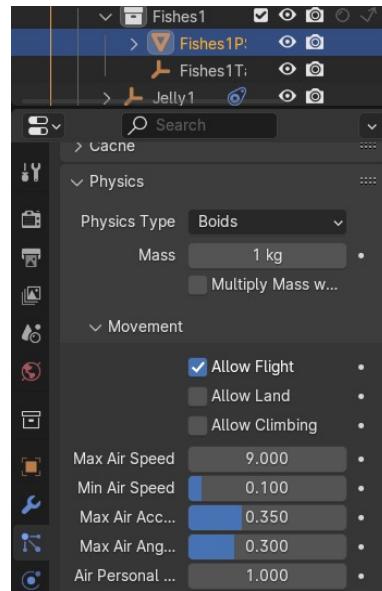
- **Rain**

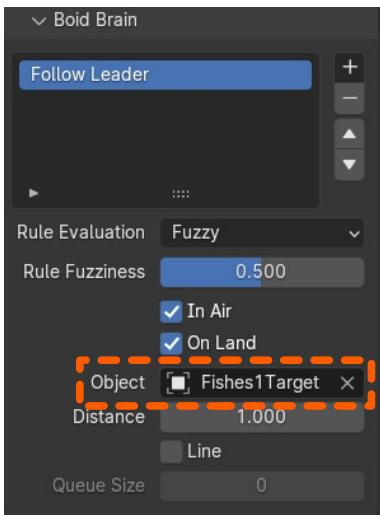
Deszcz... Kropla – *icosphere* z wyciągniętym górnym *verticle*, *shader* – **Glass, Shade Smooth**.



- **Fishes**

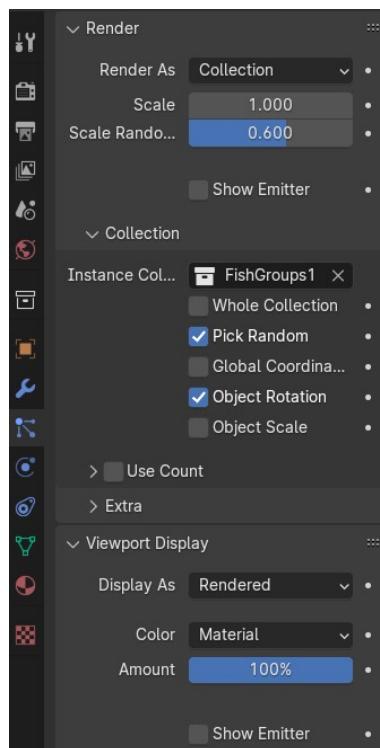
Grupa ryb1. Type – **Boids**, Render As – **Collection**.





Field Weights	
Effector Colle...	Collection
Gravity	0.000
All	1.000
Force	1.000
Vortex	0.550
Magnetic	1.000
Harmonic	1.000
Charge	1.000
Lennard-Jones	1.000
Wind	0.000
Curve Guide	1.000
Texture	1.000
Fluid Flow	1.000
Turbulence	1.000
Drag	1.000
Boid	1.000

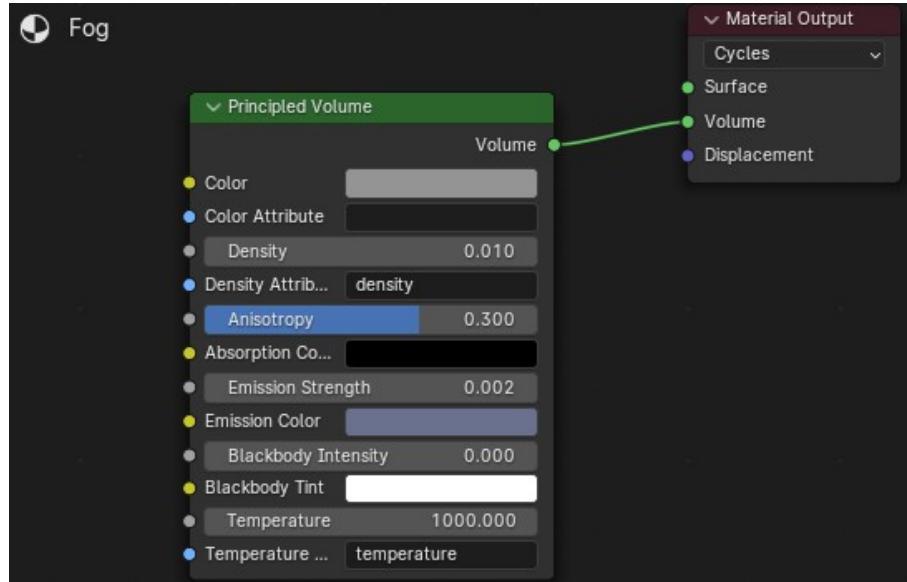
Render > Show Emitter wyłączyć.



5. Materiały

- **Mgła**

Ten sam *shader* co i dla wody, tylko inny kolor, wartości i włączona emisja.

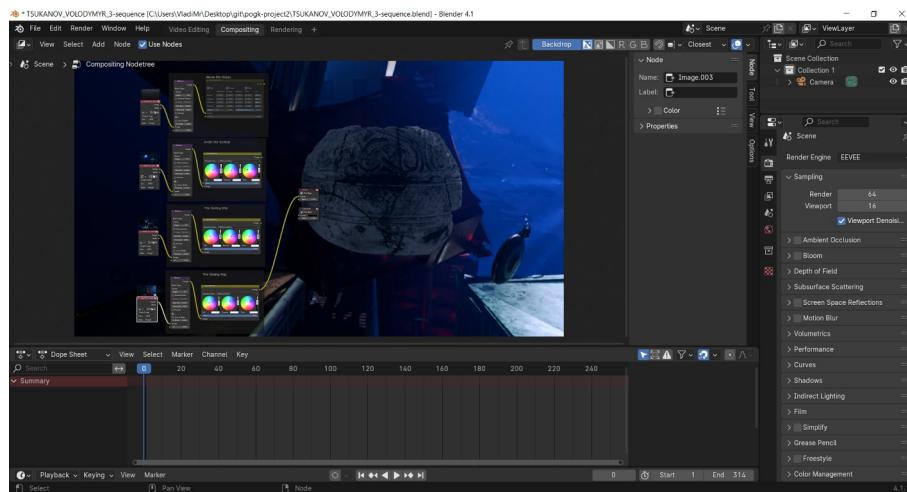


6. Renderowanie

Ulubiony etap (¬‿¬). Żeby nie czekać wieczność zwiększymy **Noise Threshold** do 0.5, co ogranicza renderowanie do 32spl, spodziewamy teraz na *denoiser OptiX*. Włączenie **Perfomance > Persistent Data** zmniejsza czas na przygotowanie przed kolejnymi klatkami (jeśli nie dużo VRAM to pozostawić w spokoju). **Use Spatial Splits = on**.

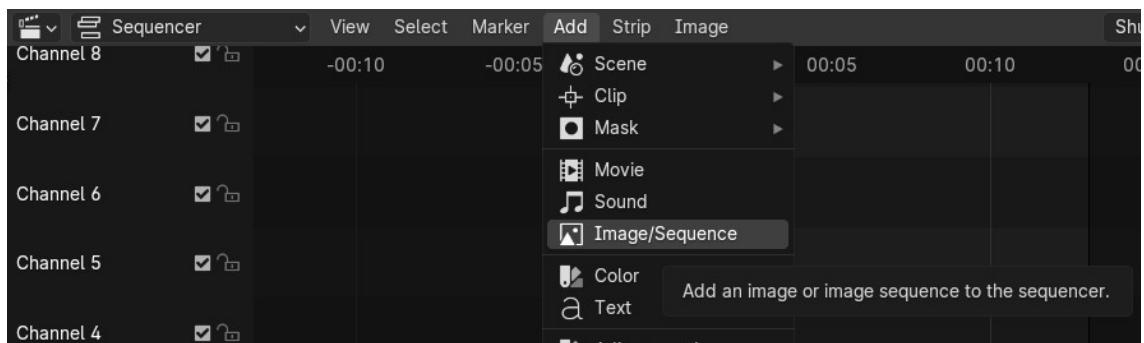
7. Compositing

Tym razem jednej sesji renderowanie nie wystarczy, robimy *post processing* redagujemy kolor i ostrość obrazków.



8. Video Editing

Importowanie wyrenderowanych *framów*.



Kolejne trzecie ostatnie finalne renderowanie obrazków do wideo.

