Interacció i Disseny d'Interfícies: Activitat 3

19 de maig de 2025

Instruccions

1. Has de partir del codi que tens a `activitat3.tgz` (el podeu trobar en el Campus digital). Has de desplegar aquest arxiu en un directori on disposaràs tots els fitxers amb els que has de treballar.

2. Els exercicis que es demanen només requereixen canvis a la classe *MyGLWidget* (.cpp i .h), als *shaders* i al fitxer *MyForm.ui* usant el designer. No has de modificar cap altre fitxer dels que se't proporcionen.

3. El codi que lliuris ha de compilar i executar correctament. Si no compila o dóna error d'execució, l'avaluació de l'exercici serà un 0, sense excepció.

4. Per a fer el lliurament has de generar un arxiu TGZ que inclogui tot el codi del teu exercici i que es digui `activitat3_NIF.tgz`, on substituiràs `NIF` pel teu número de NIF amb la lletra inclosa.

   Per exemple, l'estudiant amb NIF 12345678Z (des d'un terminal en el que s'ha col·locat dins del directori `activitat3`) farà:

   ```
   make distclean
   tar zcvf activitat3_12345678Z.tgz *
   ```

   És important fer el '`make distclean`' per a esborrar els arxius binaris generats; que el DNI sigui el correcte (el teu); i que hi hagi el sufix .tgz.

5. Has de lliurar l'exercici usant la tasca corresponent del Campus digital abans del **dilluns 1 de juny de 2025** a les 23:55.

Enunciat

El codi esquelet proporcionat dibuixa d'una escena composta de dos elements visibles i un ocult (Figura 1):

- Un fons marí blau
- Un submarí groc
- Un cub que representa el mar (no es mostra a l'escena inicial)

Es donen ja implementats els mètodes que construeixen els VAOs i VBOs dels models (`creaBuffersXXXXXX()`) i els mètodes que creen les seves TGs.

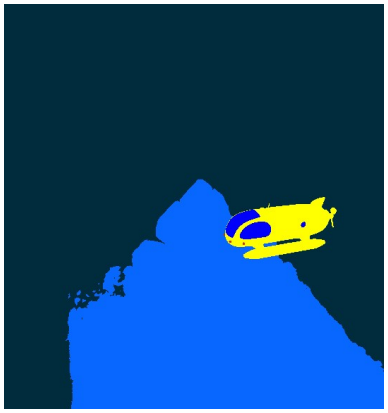


Figura 1: Escena inicial.

La càmera està calculada de manera arbitrària, de forma que permet veure el conjunt. També es dona implementat el gir de la càmera per coordenades euler mitjançant el mouse.

Per a resoldre aquesta activitat cal desenvolupar els següents punts:

1. *(0 punts)* Familiaritzeu-vos amb el codi proporcionat.
2. *(0.25 punts)* Modifiqueu el codi de forma que les tecles W / S permeten moure el submarí a l'eix **Z**, A / D a l'eix **X** i ↑ / ↓ a l'eix **Y**.
3. *(1.75 punts)* Programeu el model de Phong al **Fragment Shader**

   El color de la llum ambient és (0.1,0.1,0.1) i s'ha de passar com a `uniform` des del codi.

   Hi haurà un focus d'escena de llum blavosa ((0.1,0.6,0.5) situat a la posició (0, 50, 0) en el sistema de coordenades de l'escena (SCA o world coordinates). Cal passar les coordenades de la llum i el color de la llum com a `uniforms` L'efecte aconseguit per la il·luminació es mostra la figura 2.

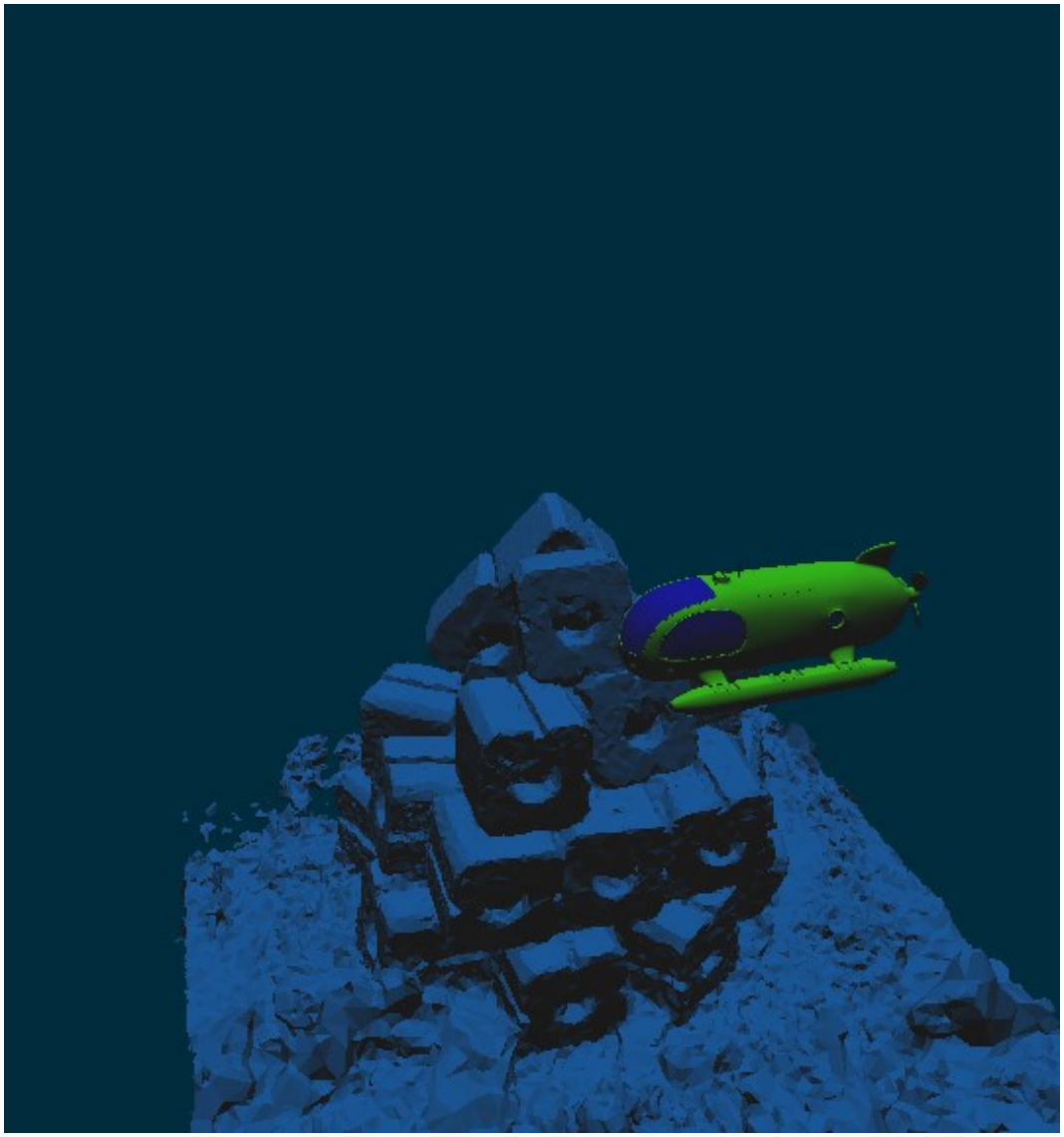


Figura 2: Escena amb il·luminació de Phong

4. *(0.5 punts)* Creeu un element d'inferfície adequat per modificar la intensitat de la llum del focus. La variació consistirà simplement en multiplicar el color de la llum per un factor escalar f de 0 a 1.

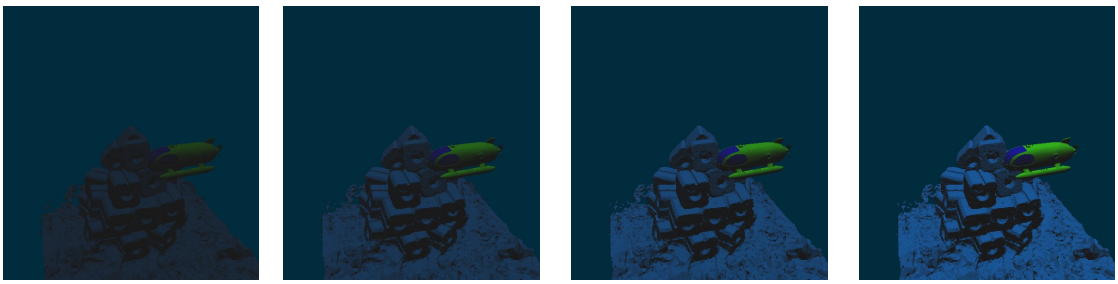


Figura 3: $f = 0.25$ $f = 0.5$ $f = 0.75$ $f = 1$

5. *(2 punts)*

   Ara s'afegiran 4 llum a l'escena, que es correspondran a les 4 llums que porta el submarí a la part frontal.

Les coordenades de les llums en el SCM (sistema de coordenades de model, és a dir, tal i com es carrega des del .obj) són les següents:

- (0.44, 0.85,3.47);
- (0.88, 0.85,2.83);
- (-0.95, 0.85,2.83);
- (-0.45, 0.85,3.47);

El color de les llums és inicialment blanc (1,1,1).

Les coordenades del les llums es passaran com a uniforms als shaders usant el sistema de coordenades que considereu pertinent, idealment en forma d'array de `vec3`.

Donat que el fet de posar 4 llums més ens "crema" l'escena per excés de luminositat, simularem que les llums tenen un abast molt curt. Per a fer-ho, si la distància entre el punt de llum i el punt il·luminat és major 0.5, simplement ignorem la llum. Altrament, procedim a calcular la il·luminació estàndard pel focus. Vegeu la Figura 4 per veure l'efecte dels 4 focus limitats en distància.

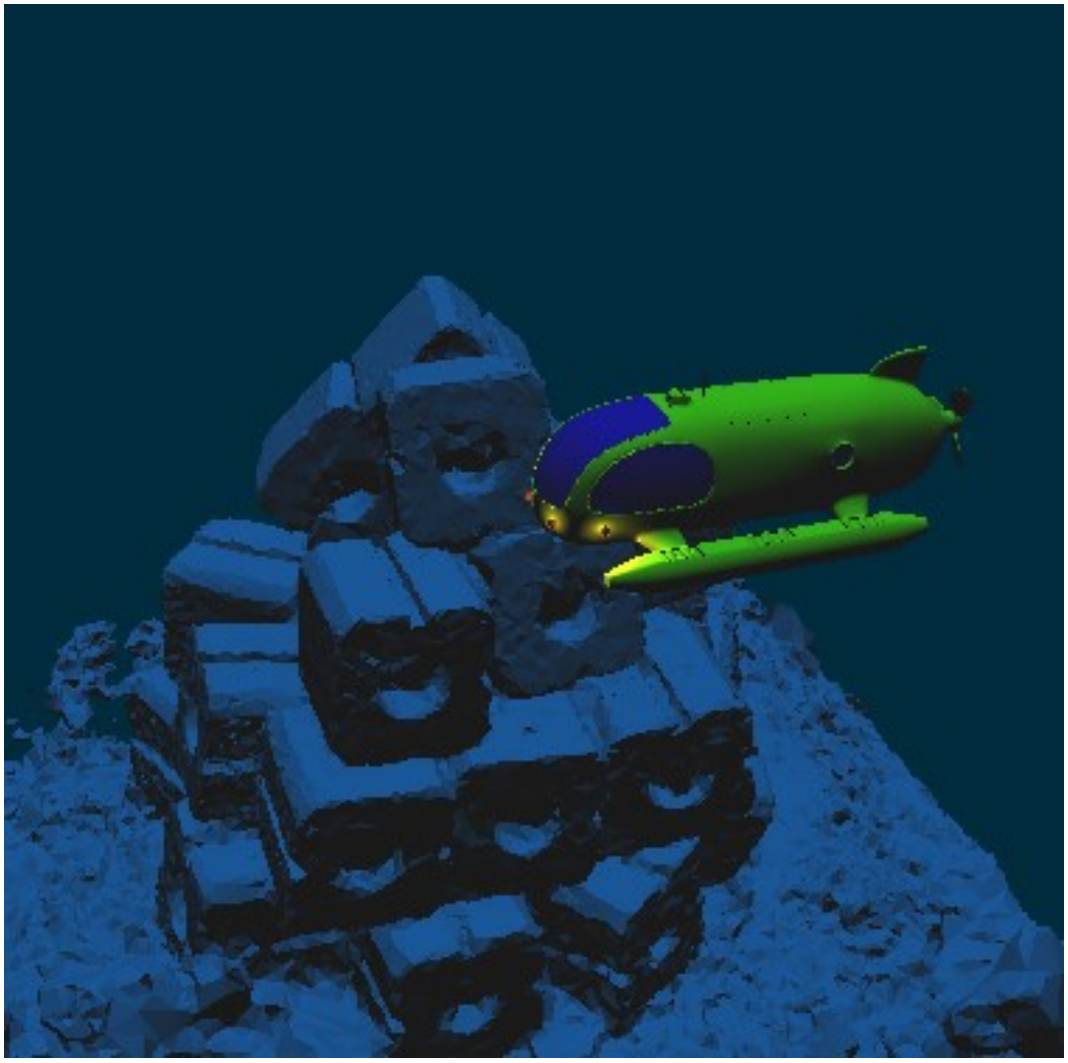


Figura 4: La il·luminació dels focus, amb un abast molt curt.

6. *(2 punts)*

   A continuació cal millorar la il·luminació dels llums de submarí. Afegirem, a banda de la llum a curta distància, un efecte de `spotlight` o focus. La idea del focus és que fem el càlcul de la llum habitual, però només per aquells punts dins d'una zona cónica, amb la

cúspide al punt de llum. Vegeu la Figura 5 com és calcula β, que és el factor pel qual hem de multiplicar el valor obtingut amb el model de Phong per simular l'efecte de focus.

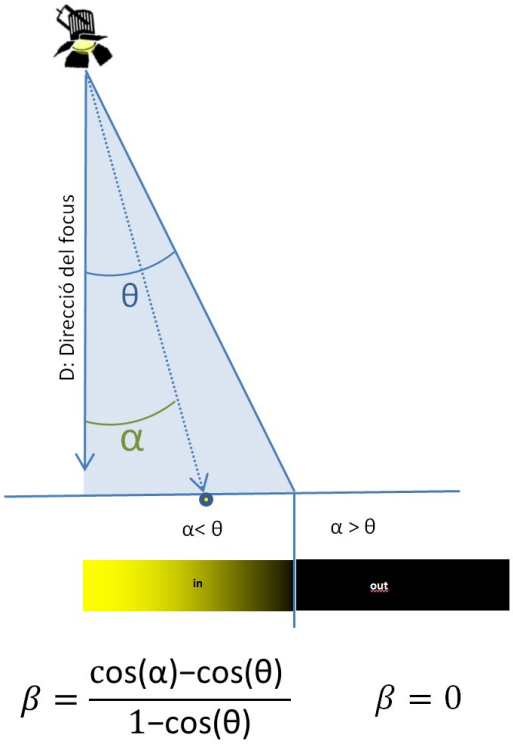


Figura 5: Com es calcula el factor de modificació de llum d'un focus simple.

Però quina és la direcció de cada focus? Farem servir com a direcció el vector que uneix l'antena amb la posició de la llum vegeu la Figura 6.

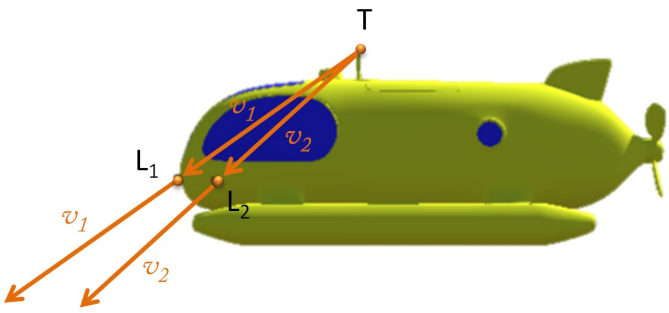


Figura 6: Diagrama que mostra com es calculen les direccions de les llums.

La posició en SCM (al model) del punt de l'antena és (2.44, 1.37, 1). Amb aquests vectors, ja podem aconseguir fàcilment el $cos(\alpha)$ donat que també sabem calcular el vector des de la llum al vèrtex il·luminat i podem usar el producte escalar. La Figura 7 mostra el resultat esperat, es poden apreciar els 4 focus a terra.

Useu una apertura de conus de $\theta = 8$ graus per obtenir el mateix efecte que la figura.

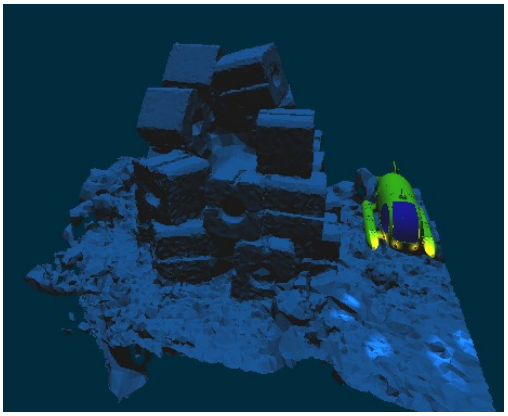
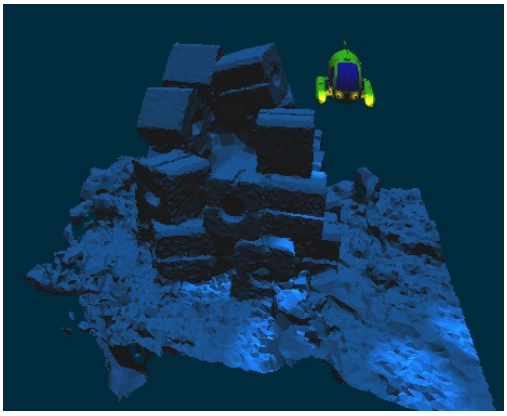


Figura 7: Imatge amb els focus actius (vista frontal). Els focus s'aprecien al terra.



Vista lateral. Quan pugem els focus es fan més grans al terra.

7. *(1.5 punts)* Afegiu elements d'interfície gràfica que permetin modificar el color de cadascun dels focus, i apagar-los i encendre'ls de forma individual.

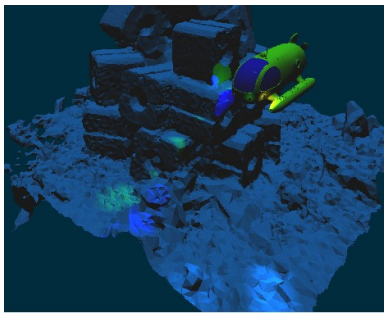


Figura 8: Tutti-colori focus.

8. *(0.5 punts)* Incorporareu a continuació un efecte de distancia molt senzill. Cal barrejar el color resultant de la il·luminació Phong amb el color negre. Com més lluny, més predomina el negre, com més aprop més predomina el color original. Això farà que els objectes llunyans s'enfosqueixin. GLSL ens ofereix la funció `mix` per fer barreges de colors. Cal passar-li un factor entre 0 i 1 que ens diu si ens acostem més a un color o a l'altre. La idea és que per a Z (en SCO) menors de 8 el color sigui l'original, per distàncies majors a 15 ja ha de ser negre, i entre 8 i 15 cal interpolar. Useu la funció `smoothstep` a tal efecte.

   La Figura 9 mostra el resultat esperat.

   Recordeu que quan es mogui el submarí, les llums cal desplaçar-les adequadament.

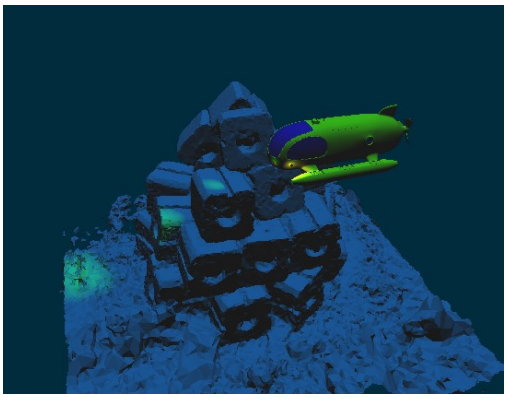
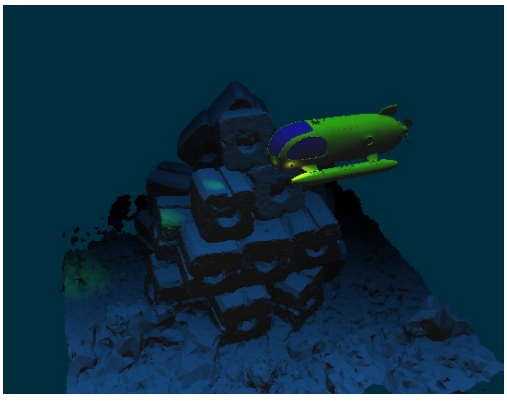


Figura 9: Original.



Amb efecte de profunditat.

9. *(1 punts)* Ara farem que els vidresdel submari siguin translúcids. Per aconseguir-ho seguirem els següents passos:

   - Activar el mode de blending a openGL amb `glEnable(GL_BLEND)`; dins de `initializeGL()`.
   - La transparència ens obliga a renderitzar primer els objectes sólids i després els objectes translucids. La idea és que quan es dibuixa el translúcid sobre el sólid, es barregen els colors.
   - La barreja s'ha de fer usant el canal alfa(a) dels colors: $(r, g, b, \mathbf{a})$. Al fragment shader forçarem una $a = 0.7$ per la part translúcida (finestres) i una $a = 1$ per la part sólida.
   - Farem servir un truc :-) en el shader per saber si estem a una part sólida o no, assumirem que tots els vidres tenen com a component difusa el color (0,0,1) (blau pur).
   - Al `paintGL()` pintarem dos cops el submarí. Haureu de passar un uniform per diferenciar la primera de la segona vegada que el dibuixem.
   - El primer cop que dibuixem el submari només pintarem la part sólida (descarteu els colors blaus), i activarem al codi c++ el mode de barreja per defecte : `glBlendFunc(GL_ONE, GL_ZERO);`
     El que fa és simplement pintar l'objecte de forma normal.
   - El segon cop que pintem el submarí només mostrarem les finestres i descartarem la part sólida. Com que ara tots els elements són translúcids i $a = 0.7$, hem d'activar el mode de barreja al c++:
     `glBlendFunc(GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA);`
     Això fa que el color final sigui la barreja del translúcid i del solid ja pintat anteriorment en el framebuffer.

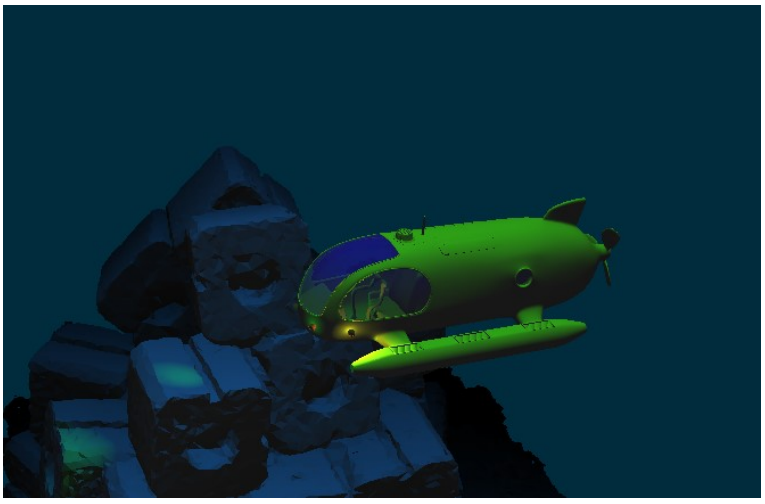


Figura 10: Finestres transparents.

10. *(0.5 punts)* Afegiu a l'escena l'objecte mar (`VAO_models[SEA]`), usant la funció `modelTransformSea()` per inicialitzar les matrius de TG. Veureu que dibuixa un prisma rectangular que conté, i oculta, tota l'escena. Caldrà que feu el prisma translúcid ($a = 0.2$) (recordeu canviar el mode de blending a c++). Feu un *flag* uniform per avisar al shader que ha de pintar el mar.

    Per fer el mar una mica més interessant, afegirem soroll. Al fragment shader teniu la funció `noise`, que genera un senyal aleatori entre 0 i 1 passant dues coordenades (podeu provar diferents combinacions de x i y multiplicades per valors al vostre gust). Barregeu el color de l'aigua amb color blanc (`mix`) usant el valor aletori.

    Jugueu amb els valor fins obtenir l'efecte desitjat, mireu la Figura 11 com a referència, però podeu proposar qualsevol variació estètica.

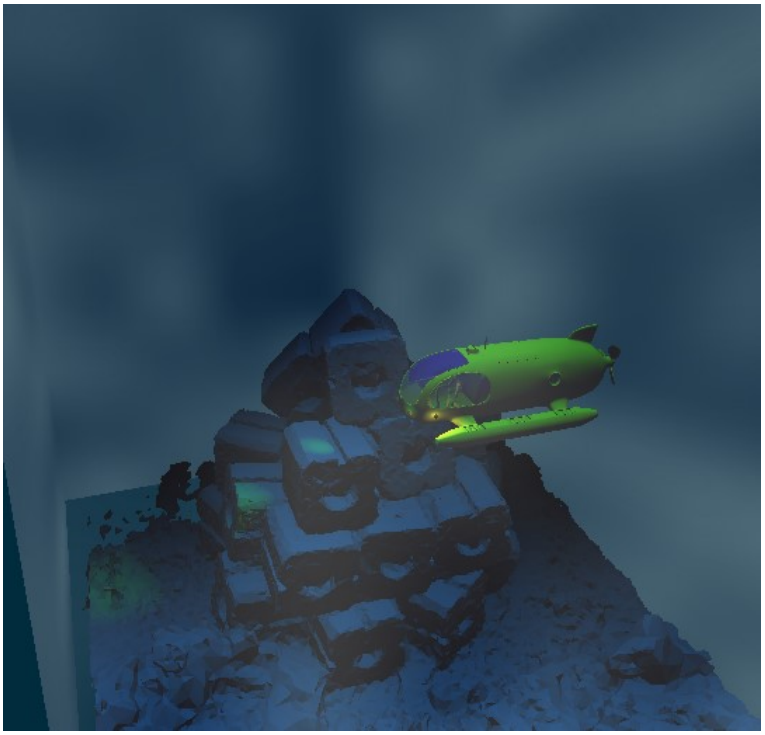


Figura 11: El mar.

11. *(1 punts / **EXTRA!**)* El toc final és simular que, en presència d'aigua, el camí de la llum dels focus també seria visible, tal i com es veu a la Figura 12 .

Figura 12: Els camins dels focus ara són visibles !

Per a fer-ho ens servirem de les cares del prisma que representen el mar. Com que sabem que sempre contindran l'escena, podem aprofitar per dibuixar el rastre dels focus sobre les cares del prisma. És un truc imaginatiu doncs realment "dibuixem"el rastre del focus sense tenir cap element geomètric que el representi. Per aconseguir-ho, des del

shader projectarem a la pantalla de projecció de la càmera (Figura 13):

- la posició del focus
- un punt en la trajectoria cap on apunta el focus
- el punt que estem il·luminant .

Això es fa simplement passant a coordenades NDC i descartant la component **Z**, cosa que ens deixarà en un món 2D (x,y). Recordeu que per arribar a les NDC cal agafar les clipping coordinates i fer la divisió de perspectiva. A partir d'aquí, hem de fer els mateixos càlculs que en 3D per saber si el punt de llum queda dins del conus del focus, usant la mateixa fórmula per obtenir β que a l'exercici 6. Per acabar, sumarem al color que li pertocaria al fragment un factor $\beta[color\_focus]$.

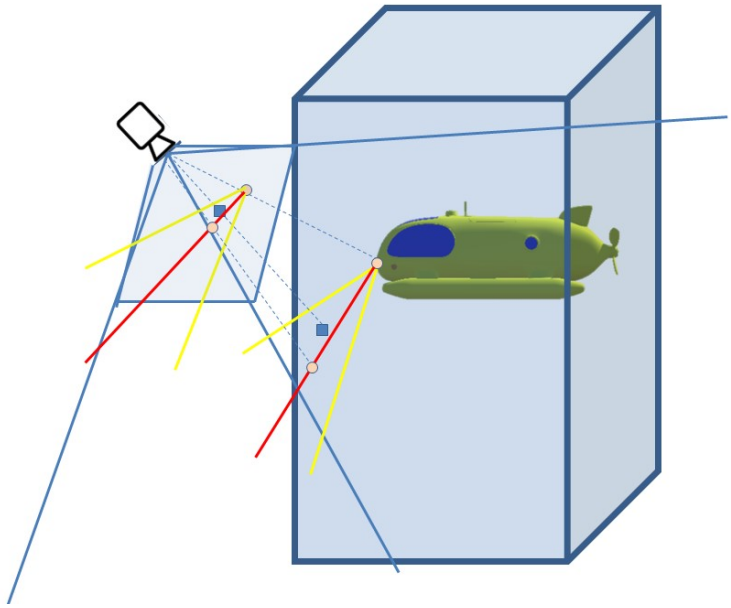


Figura 13: Projectem els punts de referència al pla de projecció de càmera per fer els càlcus del focus en 2D. El fragment que estem pintant és el representat amb el rectangle blau, els punts rodons representen el focus de llum i un punt en la direcció del focus.