

# Interacció i Disseny d'Interfícies: Activitat 3

20 de maig de 2024

## 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 *A3GLWidget* (.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 **dimarts 4 de juny de 2024** a les 23:55.

# Enunciat

El codi esquelet proporcionat dibuixa d'una escena composta de diferents elements(Figura 1):

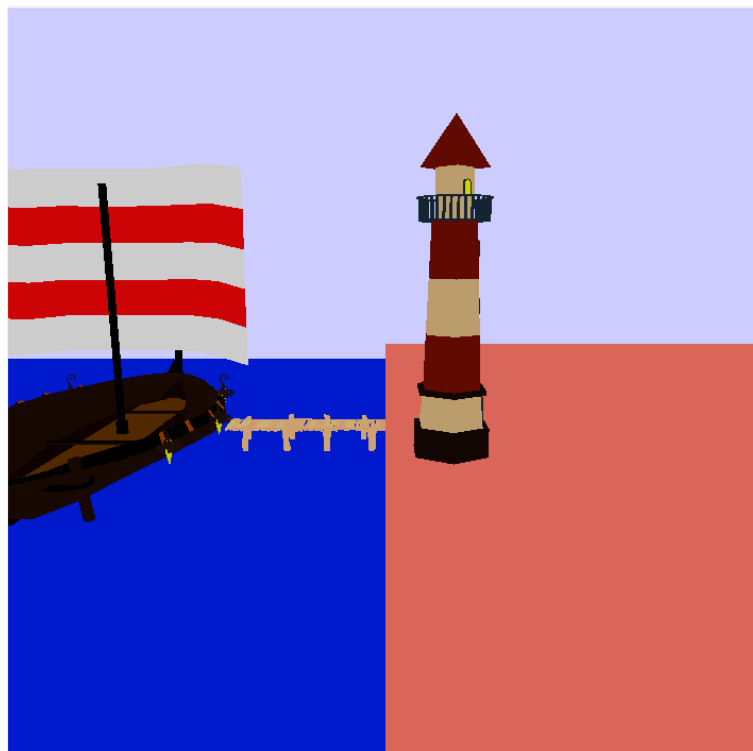
- Un far
- un moll
- un vaixell (es pot moure amb les tecles W / S )
- un cub de terra
- un cub de mar

Es donen ja implementats els mètodes que construeixen els VAOs i VBOs dels models i els mètodes que creen les seves transformacions.

El codi font està repartit en dos classes: `A3GLWidget` i `MyGLWidget`, la primera és filla de la segona. `MyGLWidget` conté la majoria d'inicialitzacions de buffers, funcions de càlcul de TG, inicialització de càmera i gestió del mouse. El vostre codi haurà d'estar dins de `A3GLWidget`, ampliant les funcionalitats de `MyGLWidget`, essencialment amb la funció `paintGL()` i totes les funcionalitats que us facin falta per aconseguir els requisits de l'enunciat. Recordeu que des de `A3GLWidget` podeu accedir a tots els camps de `MyGLWidget` (per herència).



**IMPORTANT:** Només podeu modificar els shaders (`.frag` i `.vert`), els arxius `A3GLWidget.h` i `A3GLWidget.cpp`, i el fitxer `MyForm.ui`.



**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 es demana el següent:

1. (0.25 punts) Modifica el fragment shader de forma que la il·luminació de l'escena sigui estrictament la component ambient. El color de la llum ambient és (0.1,0.1,0.1) i s'ha de passar com a `uniform` des del codi.
2. (1.25 punts) Completa el model de Phong al **Fragment Shader** afegint la component difusa i l'especular. Hi haurà un focus d'escena en representació del sol. Serà de llum blanca tènue (0.6,0.6,0.6) i estarà situat a la posició (0, 40, 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` al shader. L'efecte aconseguït per la il·luminació es mostra la figura 2.

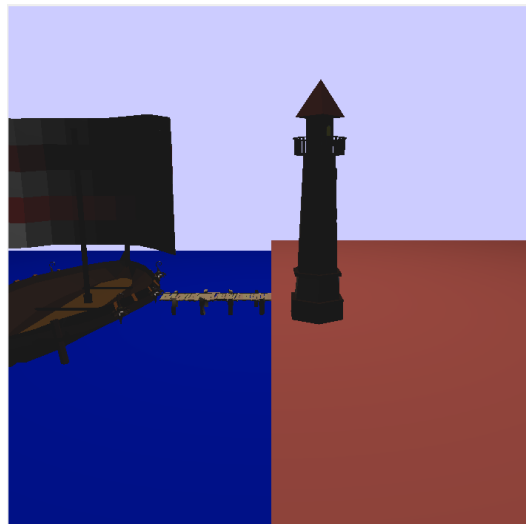
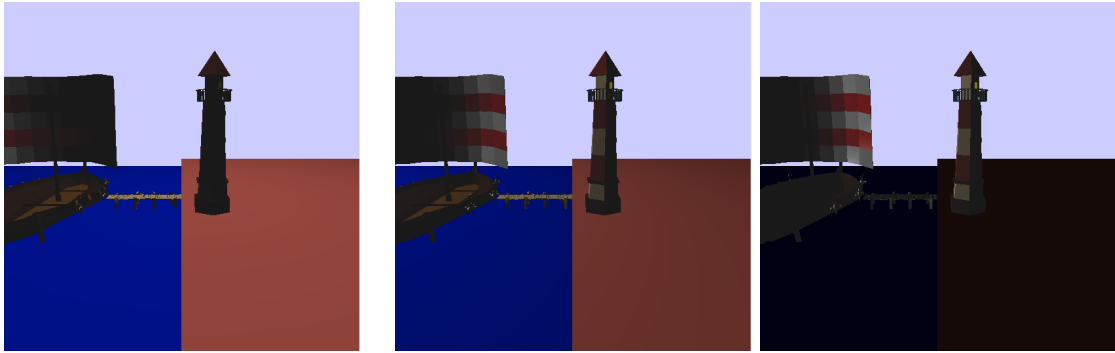


Figura 2: Escena amb il·luminació de Phong



- Verifica que passes la *Normal Matrix* com a `uniform`.
- Recorda que es més eficient passar com a `uniform` la coordenada de llum directament en SCO.

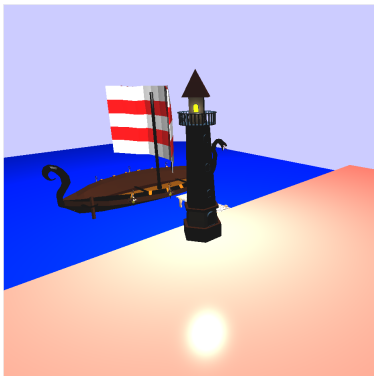
3. (0.5 punts) Aconseguiu que el sol descrigui una ruta semicircular, sortint de l'Est (-40,0,0) a les 8h, arribant al punt àlgid (0,40,0) a les 14, i posant-se a l'Oest (40,0,0) a les 20h. El punt de partida de l'escena seran les 14h, i l'usuari amb les tecles del cursor (amunt/avall) podrà sumar/restar 1 hora, sense excedir els límits [8h-20h].



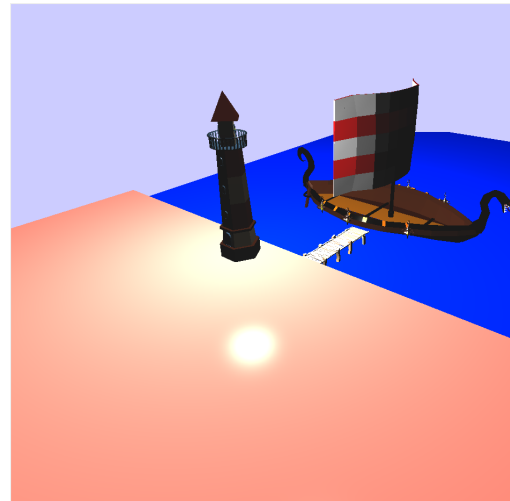
**Figura 3:** El sol a es 14h (esquerra), 11h (centre) i 8h (dreta).

4. (1 punts) Creeu elements d'interfície adequats per a visualitzar l'hora actual i poder modificar-la. Manteniu el sincronisme amb les tecles.
5. (1 punts) El far té dos focus de llum, que posarem just davant de les finestres del segment superior del far. El dissenyador gràfic ens ha dit que estan situats a les coordenades següents (coordenades de model):
  - F1: (0.363,4.695,0.357)
  - F2: (-0.357,4.695,-0.348)

Passeu les coordenades dels focus al fragment shader (useu el sistema de coordenades adequat), i implementeu la il·luminació. Teniu el resultat a la figura següent:



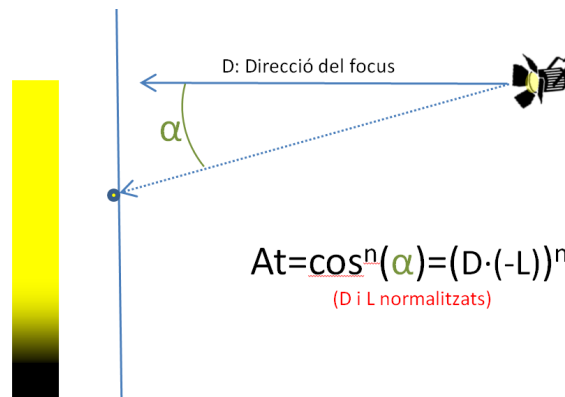
**Figura 4:** Far engegat, la finestra està il·luminada.



**Figura 5:** El costat sense finestra és fosc....però d'on surt aquesta taca especular a terra?

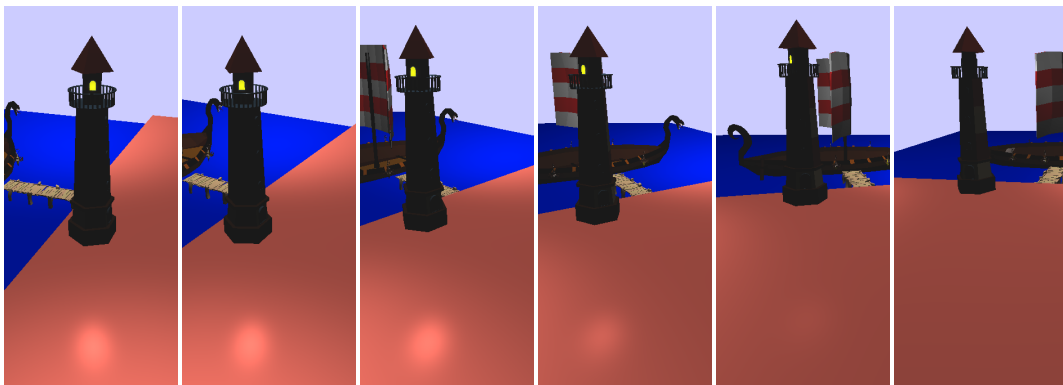
6. (1 punts) Feu girar amb el teclat ( **A** / **D** ) el segment superior del far, on estan els focus. Evidentment els focus també han de girar consistentment.

7. (1 punts) Ja em vist que els focus del far realment són fonts de llum puntual que il·luminen en totes direccions. Anem a convertir-los ara en focus de llum dirigits (*spotlights*). El càlcul de la il·luminació per un spotlight és fa seguint el model de Phong, però multiplicant la component difusa i l'especular per un factor que atenua la llum quan el punt il·luminat està lluny de la direcció apuntada per la llum. Aquest component d'atenuació,  $At$ , es calcula com el cosinus de l'angle que formen la direcció del focus amb la del raig de llum des del focus cap al punt il·luminat. Podem elevar-lo a una potència arbitrària per fer més petit o gran el focus, a l'exemple s'ha usat l'exponent  $n = 4$ .

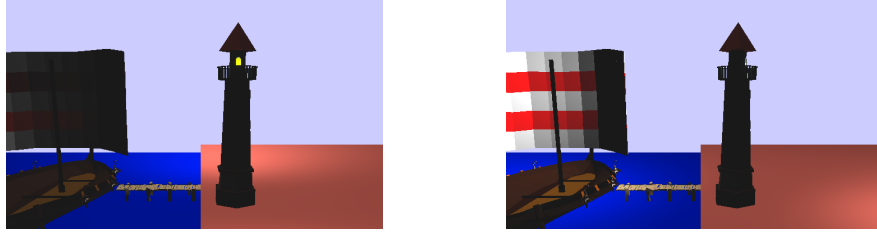


**Figura 6:** Càlcul de l'atenuació de l'spotlight

Com que els dos focus són diametralment oposats, per calcular la direcció de cada focus podeu usar el vector que els uneix,  $Dir_{F1} = Pos_{F1} - Pos_{F2}$  o viceversa, segons si volem la direcció del primer o del segon focus respectivament. Si ho feu correctament aconseguireu el resultat següent:



**Figura 7:** Detall de far en mode focus. La llum va minvant a mesura que ens separem de la línia d'orientació del focus.





**Figura 8:** Detall de far en mode focus. S'aprecia l'efecte també sobre la vela, que s'ilumina quan és apuntada pel far.

8. (2 punts) El pas següent és afegir llums a les torxes del vaixell. De nou l'artista 3D ens dona les coordenades on estan situades:

- (-7.39, 1.95,-6.68)
- (-9.95, 1.95,-0.56)
- (-7.47, 1.95, 5.64)
- ( 4.38, 1.95, 5.26)
- ( 6.68, 1.95, 0.38)
- ( 4.15, 1.95,-6.97)

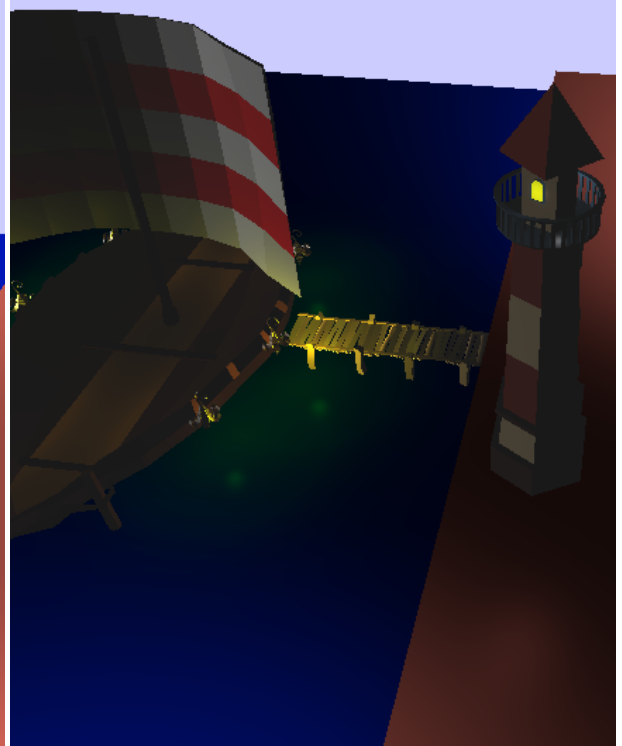
Volem col·locar llums grogues pures en aquests punts. No cal passar cada llum en un `uniform` individual, les podeu passar totes de cop com un únic array: `uniform vec3 posFocusTorxes[6];`

Per evitar 'cremar' la imatge amb tanta llum, limitarem el rang de la il·luminació de les torxes. A tal efecte calcularem la distància( $d$ ) de la torxa al punt il·luminat. Si  $d < 3$ , il·luminem normal, si  $d \geq 3$  multipliquem el resultat de Phong per un factor d'atenuació exponencial amb la distància:  $At = e^{-(d-3)}$ .

Recordeu que el vaixell es pot moure amb les tecles  /  (ja està programat), cal que les llums acompanyin al vaixell.



**Figura 9:** Il·luminació de les torxes, a les 14h.



**Figura 10:** De nit es veu millor.

9. (1 punt) Modifiqueu la interfície gràfica per tal que es pugui seleccionar:

- Si cada torxa està encesa o apagada (individualment)
- el color de la llum de les torxes (el mateix per a totes).

10. (0.75 punts) Shader del mar.

Finalment, ens agradaria crear un efecte especial sobre el mar per simular onades. En comptes d'usar les normals de l'objecte en els càlculs d'il·luminació, les generarem nosaltres a partir d'una funció. Els físics han modelat funcions d'ona i han derivat expressions per avaluar les seves normals en l'espai 3D. Al shader trobareu la funció `WaveNormal(...)` la qual, donades unes coordenades d'escena (SCA), ens retorna la normal que li correspondria a l'ona en aquell punt. Podeu ajustar diversos paràmetres de l'ona (essencialment la direcció, expressada com a vector en el pla xz, la longitud d'ona, i el temps).

**IMPORTANT:** Tingueu molt present que:

- Les normals que obtenim de la funció estan en SCA, però la il·luminació treballa amb SCO. Per tant, caldrà que la passeu els vectors normals a coordenades SCO multiplicant per la VM.
- Només aplicarem aquesta manipulació de normals al fragments que corresponen a la superfície del mar. Com els identificarem? Doncs usant un flag uniform per indicar quan dibuixem el mar, i detectant si la normal en coordenades d'escena o aplicació (SCA) apunta cap a dalt ( $y > 0$ ).

A més podeu sumar les contribucions de varies ones en diverses direccions i longituds d'ona per aconseguir efectes més realistes. Teniu un exemple a figura 11.



**Figura 11:** El mar usant una combinació de 3 ones amb longituds d'ona i direccions diferents. Trobeu la vostra combinació fagorita!

11. (0.25 punts) A la classe `MyGLWidget` trobareu l'slot `tick()`, que està connectat a un `Timer` i s'executa 10 cops per segon. Fixeu-vos el que fa. Amb aquesta informació i una miqueta de codi, podeu aconseguir que les onades estiguin animades.