	Manual de Prácticas
Secretaría/División: División de Ingeniería Eléctrica	Área/Departamento: Ingeniería en Computación


Laboratorio de Computación Gráfica e Interacción Humano Computadora

Iluminación 1

N° de practica: 07

Nombre completo de los alumnos		Firma
Arroyo Llanes Miguel Alejandro		
N° de brigada: 2	Fecha de ejecución: 05/04/2025	Grupo: 03
Calificación:	Profesor: Ing. Jose Roque Roman Guadarrama	

Elaborado por:	Revisado por:	Autorizado por:	Vigente desde:
-----------------------	----------------------	------------------------	-----------------------

	Manual de Prácticas
Secretaría/División: División de Ingeniería Eléctrica	Área/Departamento: Ingeniería en Computación

Comentario:

En esta práctica se comprendió los usos de la iluminación en código mientras a su vez se reafirmaron conocimientos como la texturización, importación de modelos y manipulación de estos dentro del código.

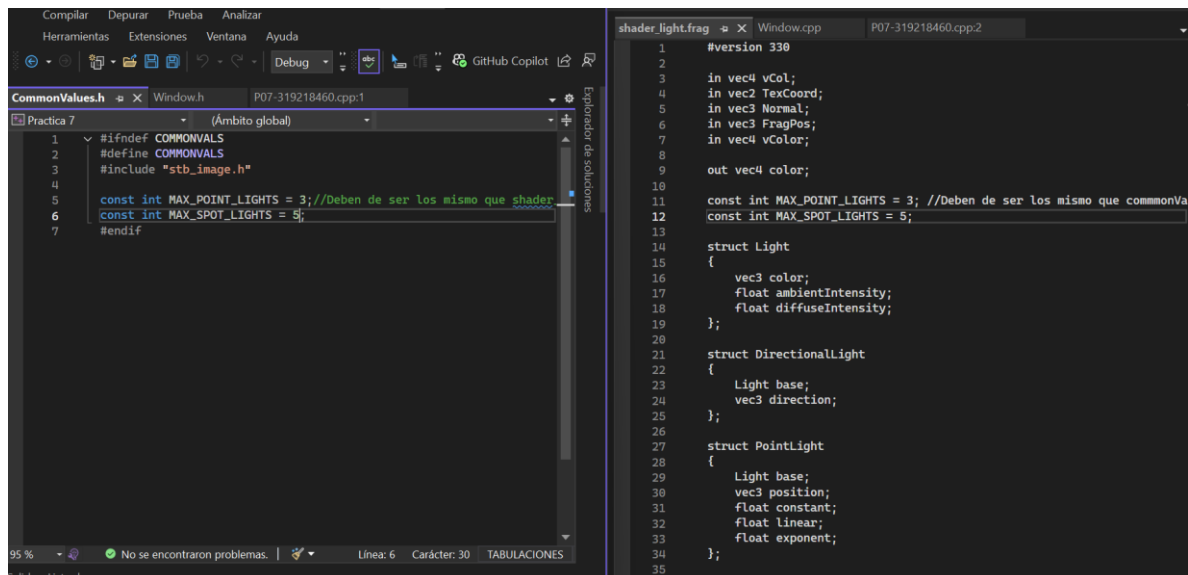
Se le implemento al helicóptero movimiento de adelante y hacia atrás con las teclas “H” e “Y”, esto aplicando lo anteriormente entendido y que ya se aplicaba con el auto (aunque ahí es con las teclas “T” y “G”).

También se le agrego una luz al helicóptero de color amarilla apuntando al piso, aunque en esta pude haber colocado la luz directamente en el origen del helicóptero, se coloco en la punta de este para que sea congruente con la ubicación usual de estos en los helicópteros, además la luz se moverá con este.

También se importo el modelo de una lampara la cual se texturizo y se implemento dentro del código, esta lampara se le coloco iluminación blanca la cual sale de el foco de la lampara y siguiendo una congruencia de su posible iluminación.

Bloque de código:


Implementación de más luces SpotLight en CommonValues.h y shader_light.frag.



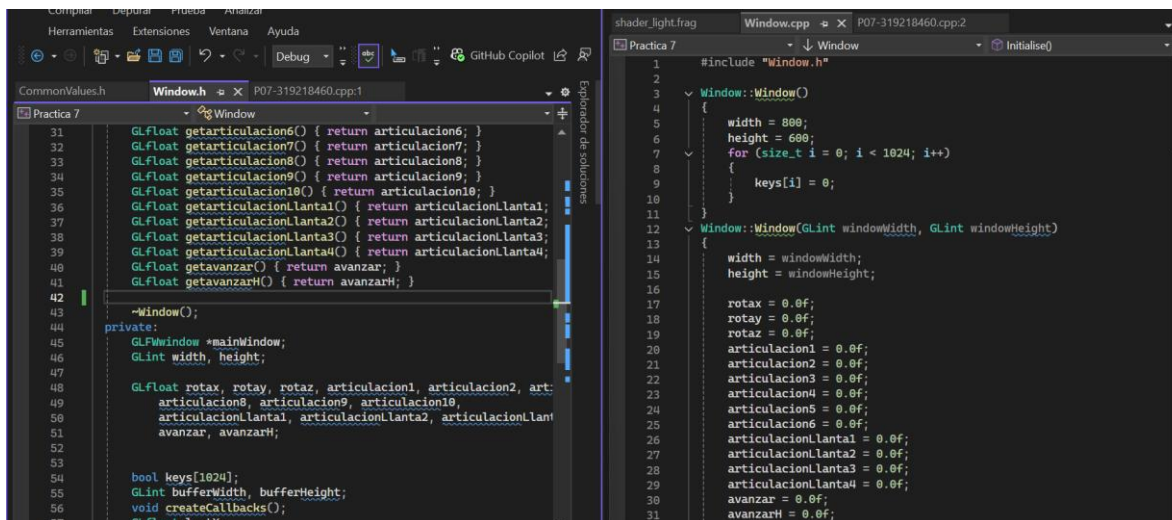
```

CommonValues.h
1 #ifndef COMMONVALS
2 #define COMMONVALS
3 #include "stb_image.h"
4
5 const int MAX_POINT_LIGHTS = 3; //Deben de ser los mismo que shader
6 const int MAX_SPOT_LIGHTS = 5;
7 #endif

shader_light.frag
1 #version 330
2
3 in vec4 vCol;
4 in vec2 TexCoord;
5 in vec3 Normal;
6 in vec3 FragPos;
7 in vec4 vColor;
8
9 out vec4 color;
10
11 const int MAX_POINT_LIGHTS = 3; //Deben de ser los mismo que commonVa
12 const int MAX_SPOT_LIGHTS = 5;
13
14 struct Light
15 {
16     vec3 color;
17     float ambientIntensity;
18     float diffuseIntensity;
19 };
20
21 struct DirectionalLight
22 {
23     Light base;
24     vec3 direction;
25 };
26
27 struct PointLight
28 {
29     Light base;
30     vec3 position;
31     float constant;
32     float linear;
33     float exponent;
34 };
35
  
```

	Manual de Prácticas
Secretaría/División: División de Ingeniería Eléctrica	Área/Departamento: Ingeniería en Computación

Controles de movimiento del helicóptero en window.h y window.cpp.



```

CommonValues.h
Window.h
P07-319218460.cpp:1

31 GLfloat getarticulacion6() { return articulacion6; }
32 GLfloat getarticulacion7() { return articulacion7; }
33 GLfloat getarticulacion8() { return articulacion8; }
34 GLfloat getarticulacion9() { return articulacion9; }
35 GLfloat getarticulacion10() { return articulacion10; }
36 GLfloat getarticulacionlanta1() { return articulacionlanta1; }
37 GLfloat getarticulacionlanta2() { return articulacionlanta2; }
38 GLfloat getarticulacionlanta3() { return articulacionlanta3; }
39 GLfloat getarticulacionlanta4() { return articulacionlanta4; }
40 GLfloat getavanzar() { return avanzar; }
41 GLfloat getavanzarH() { return avanzarH; }
42
43 ~Window();
44 private:
45   GLFWwindow *mainWindow;
46   GLint width, height;
47
48   GLfloat rotax, rotay, rotaz, articulacion1, articulacion2, art:
49   articulacion8, articulacion9, articulacion10,
50   articulacionlanta1, articulacionlanta2, articulacionlanta
51   avanzar, avanzarH;
52
53
54   bool keys[1024];
55   GLint bufferWidth, bufferHeight;
56   void createCallbacks();
57   GLfloat lastY;

```

```

shader_light.frag
Window.cpp
P07-319218460.cpp:2


1 #include "Window.h"
2
3 Window::Window()
4 {
5     width = 800;
6     height = 600;
7     for (size_t i = 0; i < 1024; i++)
8     {
9         keys[i] = 0;
10    }
11
12    Window::Window(GLint windowWidth, GLint windowHeight)
13    {
14        width = windowWidth;
15        height = windowHeight;
16
17        rotax = 0.0f;
18        rotay = 0.0f;
19        rotaz = 0.0f;
20        articulacion1 = 0.0f;
21        articulacion2 = 0.0f;
22        articulacion3 = 0.0f;
23        articulacion4 = 0.0f;
24        articulacion5 = 0.0f;
25        articulacion6 = 0.0f;
26        articulacionlanta1 = 0.0f;
27        articulacionlanta2 = 0.0f;
28        articulacionlanta3 = 0.0f;
29        articulacionlanta4 = 0.0f;
30        avanzar = 0.0f;
31        avanzarH = 0.0f;

```

```

190
191 if (key == GLFW_KEY_H) //Retroceder helicóptero
192 {
193     theWindow->avanzarH -= 3.0;
194 }
195
196 if (key == GLFW_KEY_Y) //Avanzar helicóptero
197 {
198     theWindow->avanzarH += 3.0;
199 }

```

	Manual de Prácticas
Secretaría/División: División de Ingeniería Eléctrica	Área/Departamento: Ingeniería en Computación

Importar auto y lampara.

```


55
56     //Importar Partes del auto
57     Model BaseAuto;
58     Model CofreAuto;
59     Model RuedaDerFroAuto;
60     Model RuedaIzqFroAuto;
61     Model RuedaDerTraAuto;
62     Model RuedaIzqTraAuto;
63
64     //Importar lampara
65     Model lampara;
66

```

```

228
229     //Importar auto
230     BaseAuto = Model();
231     BaseAuto.LoadModel("Models/BaseAuto.obj");
232     CofreAuto = Model();
233     CofreAuto.LoadModel("Models/CofreAuto.obj");
234     RuedaDerFroAuto = Model();
235     RuedaDerFroAuto.LoadModel("Models/RuedaDerFroAuto.obj");
236     RuedaIzqFroAuto = Model();
237     RuedaIzqFroAuto.LoadModel("Models/RuedaIzqFroAuto.obj");
238     RuedaDerTraAuto = Model();
239     RuedaDerTraAuto.LoadModel("Models/RuedaDerTraAuto.obj");
240     RuedaIzqTraAuto = Model();
241     RuedaIzqTraAuto.LoadModel("Models/RuedaIzqTraAuto.obj");
242
243     //Importar lampara
244     lampara = Model();
245     lampara.LoadModel("Models/lampara.obj");
246

```

	Manual de Prácticas
Secretaría/División: División de Ingeniería Eléctrica	Área/Departamento: Ingeniería en Computación

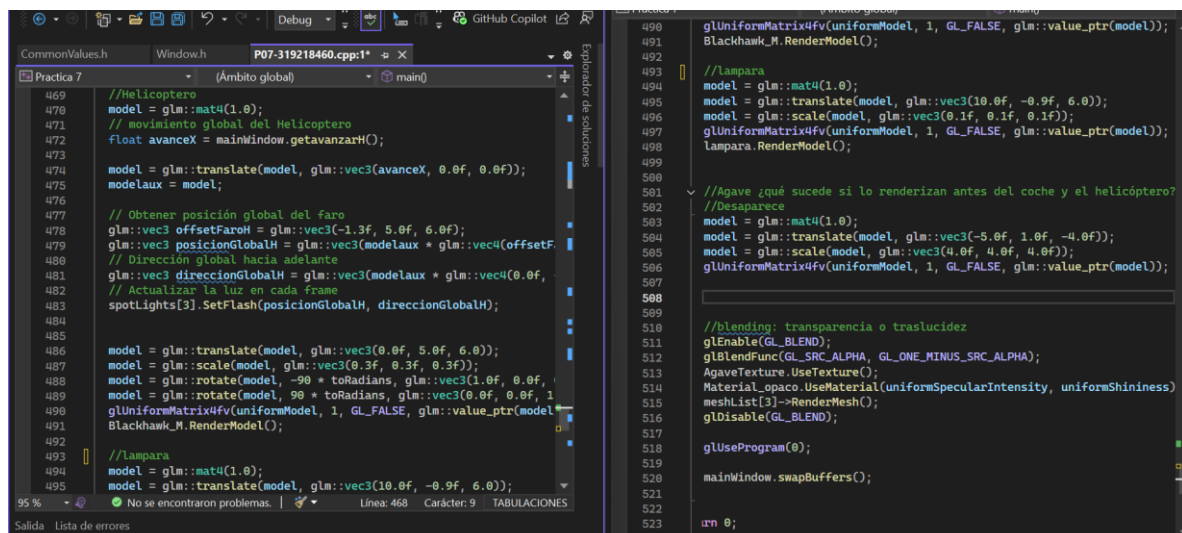
Luces de helicóptero y lampara.

```

302
303
304 //luz fija helicoptero
305 spotLights[3] = SpotLight(1.0f, 1.0f, 0.0f, //Color
306     1.0f, 2.5f, //ambient, diffuse
307     -1.3f, 5.0f, 6.0f, //posición inicial
308     0.0f, -1.0f, 0.0f, //dirección
309     1.0f, 0.0f, 0.0f, //atenuación
310     12.0f); //ángulo de luz (borde d
311 spotLightCount++;
312
313
314 //luz fija lampara
315 spotLights[4] = SpotLight(1.0f, 1.0f, 1.0f, //Color
316     0.5f, 2.5f, //ambient, diffuse
317     10.0f, 13.7f, 11.9, //posición inicial
318     0.0f, -1.0f, 0.3f, //dirección
319     0.5f, 0.0f, 0.0f, //atenuación
320     16.0f); //ángulo de luz (borde d
321 spotLightCount++;
322

```


Colocación del modelo de la lampara y helicóptero así como agregar que la luz del helicóptero se mueva con él.



```


469 //Helicoptero
470 model = glm::mat4(1.0);
471 // movimiento global del Helicoptero
472 float avanceX = mainWindow.getAvanzarH();
473
474 model = glm::translate(model, glm::vec3(avanceX, 0.0f, 0.0f));
475 modelaux = model;
476
477 // Obtener posición global del faro
478 glm::vec3 offsetFaroH = glm::vec3(-1.3f, 5.0f, 6.0f);
479 glm::vec3 posicionGlobalH = glm::vec3(modelaux * glm::vec4(offsetFaroH, 1.0f));
480 // Dirección global hacia adelante
481 glm::vec3 direccionGlobalH = glm::vec3(modelaux * glm::vec4(0.0f, 0.0f, 1.0f, 1.0f));
482 // Actualizar la luz en cada frame
483 spotLights[3].SetFlash(posicionGlobalH, direccionGlobalH);
484
485
486 model = glm::translate(model, glm::vec3(0.0f, 5.0f, 6.0f));
487 model = glm::scale(model, glm::vec3(0.3f, 0.3f, 0.3f));
488 model = glm::rotate(model, 90 * toRadians, glm::vec3(1.0f, 0.0f, 0.0f));
489 model = glm::rotate(model, 90 * toRadians, glm::vec3(0.0f, 0.0f, 1.0f));
490 glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
491 Blackhawk_M.RenderModel();
492
493 //lampara
494 model = glm::mat4(1.0);
495 model = glm::translate(model, glm::vec3(10.0f, -0.9f, 6.0f));
496
497 glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
498 Lampara.RenderModel();
499
500
501 //Agave ¿qué sucede si lo renderizan antes del coche y el helicóptero?
502 //Desaparece
503 model = glm::mat4(1.0);
504 model = glm::translate(model, glm::vec3(-5.0f, 1.0f, -4.0f));
505 model = glm::scale(model, glm::vec3(4.0f, 4.0f, 4.0f));
506 glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
507
508
509
510 //blending: transparencia o translucidez
511 glEnable(GL_BLEND);
512 glBlendFunc(GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA);
513 AgaveTexture.UseTexture();
514 Material_opaco.UseMaterial(uniformSpecularIntensity, uniformShininess);
515 meshList[3] -> RenderMesh();
516 glDisable(GL_BLEND);
517
518
519 glUseProgram(0);
520
521 mainWindow.swapBuffers();
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000

```

	Manual de Prácticas
Secretaría/División: División de Ingeniería Eléctrica	Área/Departamento: Ingeniería en Computación

Ejecución:



	Manual de Prácticas
Secretaría/División: División de Ingeniería Eléctrica	Área/Departamento: Ingeniería en Computación



Conclusión:

Se pusieron en practica todos los conocimientos adquiridos hasta el momento, esta practica me ha ayudado a comprender de una mejor manera el uso de la luz así como manipularla para que se mueva con los objetos y controlar de una mejor manera su dirección y el de donde “nace”, sin embargo a su vez lo único que se me complico más fue el controlar de donde nacía la luz de la lampara donde fue más que nada hacer prueba y error hasta que me convenciera el cómo se veía lo cual se logro al final, detalles así noto que son más complicados y se debe de encontrar una manera de controlar el punto de donde nace la luz de una forma más exacta y precisa para que tenga congruencia con por ejemplo en este caso la salida de un foco.

Bibliografía:

1. Lámpara de pie Bronx, lámpara clásica de acero. (2024, 27 mayo). Open3DModel.com. https://open3dmodel.com/es/3d-models/bronx-floor-lamp-classic-steel-lamp_349426.html