



Department of Computer
Science and Engineering

Final Project

Project Title:

Forest with Modern Objects

Name: Abdultawwab Safarji (3710933)

Submitted to:

Mr.Basim Iskandarani

Computer Graphics (CS-376)

Apr 15, 2020

1 Problem Statement

There are many ideas to put into imagination, but I would like to apply all of what I have learned so far. So, I would like to make a wide area of green trees and using the helicopter that we have created and make it more controllable. I wanted to build houses and area for parking the helicopter as well. I will use all the graphics features such as texture, animation, lighting, and 3D objects.

2 Objective of the Project

The goal of this project is to create big field of green area, houses and place for parking the helicopter as well. I will use all the graphics features such as texture, animation, lighting, and 3D objects.

.

3 Functionalities

In order to achieve the goal of the project is to use the following functions,

- Lighting
- Textuer
- Coloring
- Drawing vertexes 3D
- Animation
- 3D objects



Drawing an example

4 Code

```
1153 void InitLists()
1154 {
1155     //glLoadIdentity();
1156     float dx = BOXSIZE / 7.f;
1157     float dy = BOXSIZE / 9.f;
1158     float dz = BOXSIZE / 8.f;
1159     glPushMatrix();
1160     glScalef(50, 50, 1.0f);
1161     //glScalef(1.0f, 1.0f, rotate_x);
1162     // glRotatef(Time, -1.0f, 1.5f, -5.0f);
1163     BoxList = glGenLists(1);
1164     glNewList(BoxList, GL_COMPILE);
1165     // glTranslatef( 6, 5, 7 );
1166     // glRotatef(Time, 0., 0., 1.);
1167     glTranslatef(0.0, -10.0, -5.0);
1168     //glShadeModel( GL_FLAT );
1169     // glNormal3f( 5, 3, 6);
1170     glBegin(GL_POLYGON);
1171     glNormal3f( 1, 1, 1 );
1172     glVertex2f(0.90, 0.60);
1173     // glRotatef(Time, -1.0f, 1.5f, -5.0f);
1174     glNormal3f( 2, 2, 2 );
1175     glVertex2f(0.50, 0.120);
1176     glVertex2f(0.10, 0.80);
1177     glVertex2f(0.80, 0.10);
1178     glBegin(GL_LINE_LOOP); //GL_QUADS GL_TRIANGLE_STRIP or GL_QUAD_STRIP GL_TRIANGLES GL_QUADS GL_LINE_LOOP
1179     glColor3f(0., 0., 1.);
1180     glColor3f(2., 0., 1.);
1181     glNormal3f(0., 0., 1.);
1182     glScalef(2.0f, 2.0f, 1.0f);
1183     glVertex3f(-dx, -dy, dz);
1184     glVertex3f(dx, -dy, dz);
1185     glVertex3f(dx, dy, dz);
1186     glNormal3f(-1., 0., 0.);
1187     glVertex3f(-dx, -dy, dz);
1188     glVertex3f(-dx, dy, dz);
1189     glVertex3f(-dx, dy, -dz);
1190     glVertex3f(-dx, -dy, -dz);
1191     for (dx = dy; dx <= 6.6; dx++) {
1192         glColor3f(0., 0., 1.);
1193         for (int i = 0; i < 10; i++) {
1194             glColor3f(i, 4., 1.);
1195             glVertex3f(1, i, dy);
1196             glVertex3f(i, 1, dx);
1197             glVertex3f(2, dx, -i);
1198             glVertex3f(-i, 2, -dz);
1199         }
1200         glScalef(2.0f, 2.0f, 1.0f);
1201         glVertex3f(-dx, dy, dz);
1202         glVertex3f(dx, dy, dz);
1203         glVertex3f(-dx, dy, -dz);
1204         glVertex3f(-dx, dy, -dz);
1205         glNormal3f(0., 0., -1.);
1206         glTexCoord2f(0., 0.);
1207         glVertex3f(-dx, -dy, -dz);
1208         glTexCoord2f(0., 1.);
1209         glVertex3f(-dx, dy, -dz);
1210         glTexCoord2f(1., 1.);
1211         glVertex3f(dx, dy, -dz);
1212         glTexCoord2f(1., 0.);
1213         glVertex3f(dx, -dy, -dz);
1214         glColor3f(4., 1., 0.);
1215         glNormal3f(0., 1., 0.);
1216         glVertex3f(-dx, dy, dz);
1217         glVertex3f(dx, dy, dz);
1218         glVertex3f(dx, dy, -dz);
1219         glVertex3f(-dx, dy, -dz);
1220     }
1221     glVertex3f(-dx, dy, dz);
1222 }
```

```

902 void DoTexture(int id)
903 {
904     idTexture = id;
905     switch (id)
906     {
907         case 0:
908             Texture = Texture1;
909             isTexture = true;
910             break;
911         case 1:
912             Texture = Texture2;
913             isTexture = true;
914             break;
915         case 2:
916             Texture = Texture3;
917             isTexture = true;
918             break;
919         case 3:
920             isTexture = false;
921             break;
922     }
923 }
924
925 // main menu callback:
926
927 void DoMainMenu(int id)
928 {
929     switch (id)
930     {
931         case RESET:
932             Reset();
933             break;
934
935         case QUIT:
936             // gracefully close out the graphics:
937             // gracefully close the graphics window:
938             // gracefully exit the program:
939             glutSetWindow(MainWindow); ⚠ 'glutSetWindow' is deprecated: first deprecated in macOS 10.11
940             glFinish();
941             glutDestroyWindow(MainWindow); ⚠ 'glutDestroyWindow' is deprecated: first deprecated in macOS 10.11
942             exit(0);
943             break;
944
945         default:
946             fprintf(stderr, "Don't know what to do with Main Menu ID %d\n", id);
947     }
948 }

```

```

621 // cone
622 glPushMatrix();
623 glColor3f(0.5, 0, 1);
624 glTranslatef(0, 0, -21.5);
625 MjbSphere(1, 50, 50);
626 glPopMatrix();
627
628 // Red Sphere
629 glPushMatrix();
630 glColor3f(1, 0, 0);
631 SetMaterial(0, 1, 0, 2);
632 glTranslatef(-12, -12.0, 2.0);
633 glRotatef((float)Time * 4000, 1, 0, 0);
634 glTranslatef(12, 12.0, 2.0);
635 if (LIGHT00n)
636 {
637     glEnable(GL_LIGHT0);
638     SetPointLight(GL_LIGHT0, 3,3, 3, 2, 0, 0);
639 }
640 else
641     glDisable(GL_LIGHT0);
642 MjbSphere(1.5, 50, 50);
643 glPopMatrix();
644 // red sphere
645 glPushMatrix();
646 glColor3f(1, 0, 0);
647 glTranslatef(-12, -12.0, 2.0);
648 glRotatef((float)Time * 4000, 1, 0, 0);
649 glTranslatef(12, 12.0, 2.0);
650 MjbSphere(0.5, 50, 50);
651 glPopMatrix();
652
653 glPushMatrix();
654 draw_building();
655 draw_helipad();
656
657 // glPushMatrix();
658
659 glEnable(GL_LIGHTING);
660 glScalef(100,100,100);
661 glColor3f(0.8, 0.2, 1.0);
662
663
664 glTranslatef(15.0, 5, 10.0);
665 // glRotated(10, 0.0, 0.0, 0.0);
666 glCallList(BoxList);
667 glDisable(GL_LIGHTING);
668 // glPopMatrix();
669
670

```

```

671
672     glPopMatrix();
673     glPushMatrix();
674     glTranslatef(-40, 0, -50);
675 draw_house();
676     glPopMatrix();
677     // Blue Sphere
678     glPushMatrix();
679     glColor3f(0, 0, 1);
680     SetMaterial(0, 1, 0, 2);
681     glTranslatef(0, 0, -14.0);
682     glTranslatef(10, 10.0, -10.0);
683     glRotatef((float)Time * 4000, 1, 0, 0);
684     glTranslatef(-10, -10.0, -10.0);
685     if (LIGHT1On)
686     {
687         glEnable(GL_LIGHT1);
688         SetPointLight(GL_LIGHT1, 2, 2, 2, 0, 0, 1);
689     }
690     else
691         glDisable(GL_LIGHT1);
692     MjbSphere(1.5, 50, 50);
693     glPopMatrix();
694     // Small the blu sphere
695     glPushMatrix();
696     glColor3f(0, 0, 1);
697     glTranslatef(0, 0, -14.0);
698     glTranslatef(10, 10.0, -10.8);
699     glRotatef((float)Time * 4000, 1, 0, 0);
700     glTranslatef(-10, -10.0, -10.8);
701     MjbSphere(0.5, 50, 50);
702     glPopMatrix();
703
704     // S texture
705     glPushMatrix();
706     glShadeModel(GL_SMOOTH);
707
708     glColor3f(1, 1, 1);
709     glTranslatef(0, -10, 15);
710     SetMaterial(0, 1, 0, 2);
711     if (isTexture)
712     {
713         glEnable(GL_TEXTURE_2D);
714         glBindTexture(GL_TEXTURE_2D, tex);
715         MjbSphere(1000, 300, 300);
716
717         // glTexCoord2f(100,20);
718         glDisable(GL_TEXTURE_2D);
719     }

```

```

570
571
572     glShadeModel(GL_FLAT);
573     glColor3f(0, 1, 0);
574     glTranslatef(0, 0, 14);
575     SetMaterial(0, 1, 0, 1);
576     glutSolidTorus(3, 10, 64, 64);
577
578     // glutSolidTeapot(5);
579     // glutSolid(3,
580     ///0,/Users/safarji/Downloads/kk/worldtex.bmp
581     //
582         6;
583
584
585     glPushMatrix();
586     glColor3f(0.5, 0.2, 1);
587     glTranslatef(0, 0, -25);
588     SetMaterial(0, 1, 0, 2);
589     if (LIGHT2On)
590     {
591         SetSpotLight(GL_LIGHT2, 20, 20, 20, -1, -1, -1, 0.5, 0.2, 1);
592     }
593     else
594     {
595         glDisable(GL_LIGHT2);
596     }
597
598     glPushMatrix();
599     glShadeModel(GL_SMOOTH);
600     // glColor3f(1, 1, 1);
601     glTranslatef(0, -10, 15);
602     // SetMaterial(0, 1, 0, 2);
603     //glutSolidTeapot(6);
604     //glutSolidCone(59,50,55,50);
605     glTranslatef(0,-50, 0);
606
607     glEnable(GL_TEXTURE_2D);
608     glBindTexture(GL_TEXTURE_2D, tex0);
609     //MjbSphere(3, 50, 50);
610     glScalef(1, 0.1, 1);
611
612     glutSolidCube(300);
613
614     //glTexCoord2f(100,20);
615     glDisable(GL_TEXTURE_2D);
616     // glDisable(GL_LIGHTING);
617     glPopMatrix();
618

```



```

452 void Display()
453 {
454     // light();
455     GLuint tex;
456     glPixelStorei(GL_UNPACK_ALIGNMENT, 1);
457     glGenTextures(1, &tex);
458
459     glBindTexture(GL_TEXTURE_2D, tex); // make tex texture current
460     // and set its parameters
461
462     glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_REPEAT);
463     glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_REPEAT);
464     glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_NEAREST);
465     glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_NEAREST);
466     glTexEnvf(GL_TEXTURE_ENV, GL_TEXTURE_ENV_MODE, GL_REPLACE);
467     glTexImage2D(GL_TEXTURE_2D, 0, 3, width, height, 0, GL_RGB, GL_UNSIGNED_BYTE, Texture);
468
469
470     GLuint tex0;
471     glPixelStorei(GL_UNPACK_ALIGNMENT, 1);
472     glGenTextures(1, &tex0);
473
474     glBindTexture(GL_TEXTURE_2D, tex0); // make tex texture current
475     // and set its parameters
476
477     glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_REPEAT);
478     glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_REPEAT);
479     glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, GL_NEAREST);
480     glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MIN_FILTER, GL_NEAREST);
481     glTexEnvf(GL_TEXTURE_ENV, GL_TEXTURE_ENV_MODE, GL_REPLACE);
482
483     glTexImage2D(GL_TEXTURE_2D, 0, 3, width, height, 0, GL_RGB, GL_UNSIGNED_BYTE, Textur);
484     //glEnable(GL_TEXTURE_CUBE_MAP);
485     //glTexEnvf(GL_TEXTURE_ENV, GL_TEXTURE_ENV_MODE, GL_DECAL);
486
487     if (DebugOn != 0)
488     {
489         fprintf(stderr, "Display\n");
490     }
491
492     // set which window we want to do the graphics into:
493
494     glutSetWindow(MainWindow); ⚠ 'glutSetWindow' is deprecated: first deprecated in macOS 10.9 - OpenG...
495
496     // erase the background:
497
498     glDrawBuffer(GL_BACK);
499     glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
500
501     if (DepthBufferOn != 0)
502         glEnable(GL_DEPTH_TEST);

```

```

void Keyboard(unsigned char c, int x, int y)
{
    if (DebugOn != 0)
        fprintf(stderr, "Keyboard: '%c' (0x%0x)\n", c, c);

    switch (c)
    {
        case 'o':
        case 'O':
            WhichProjection = ORTHO;
            break;

        case 'p':
        case 'P':
            WhichProjection = PERSP;
            break;

        case 't':
        case 'T':
            if (idTexture < 3)
            {
                DoTexture(idTexture + 1);
            }
            else
            {
                DoTexture(0);
                idTexture = 0;
            }
            break;

        case 'd':
        case 'D':
            Distort = !Distort;
            break;

        case 'f':
        case 'F':
            Frozen = !Frozen;
            break;

        case 'l':
            LIGHTING = !LIGHTING;
            break;

        case '1':
            LIGHT0On = !LIGHT0On;
            break;

        case '2':
            LIGHT1On = !LIGHT1On;
            break;

        case '3':
            LIGHT2On = !LIGHT2On;
            break;

        case '4':
            LIGHT3On = !LIGHT3On;
            break;

        case '5':
            LIGHT4On = !LIGHT4On;

            break;

        case 'q':
        case 'Q':
        case ESCAPE:
            DoMainMenu(QUIT); // will not return here
            break; // happy compiler

        default:

```

5 Demo

Tips:

D= distortion

P= prospect

zero= lightings off

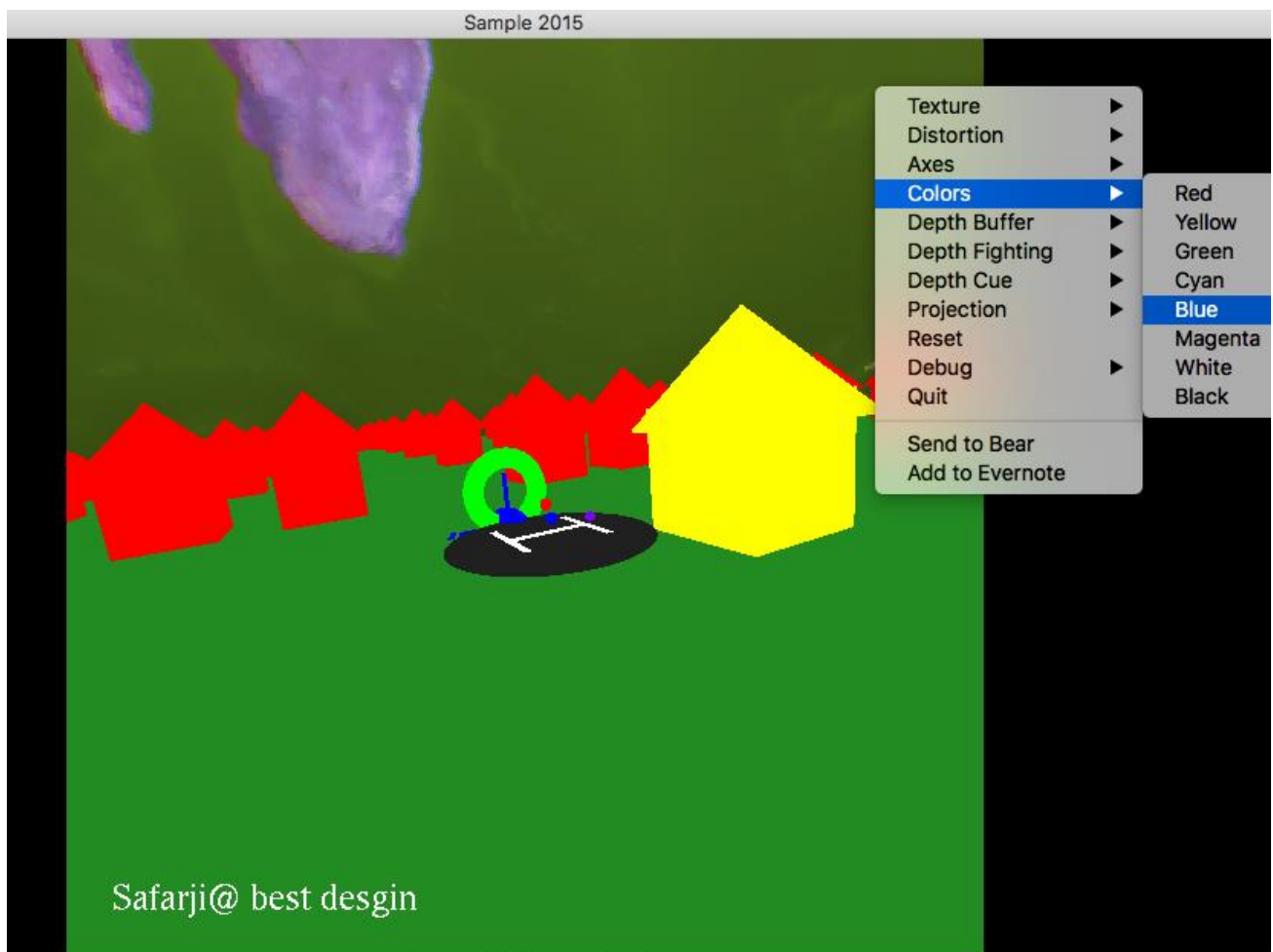
1 to 4=lights and shadow

T= multi textuers

5= sky mapping

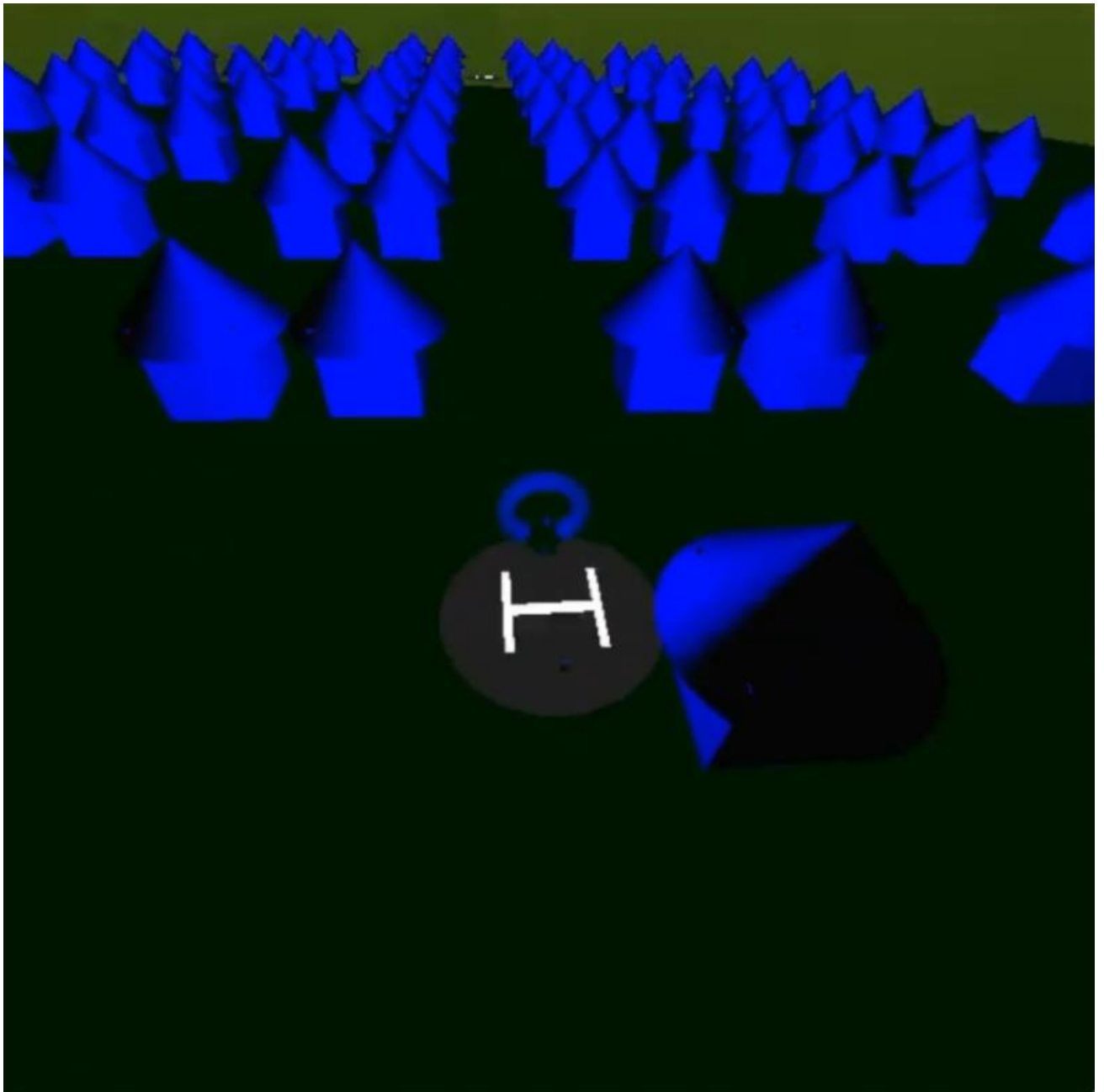
F= freeze animation

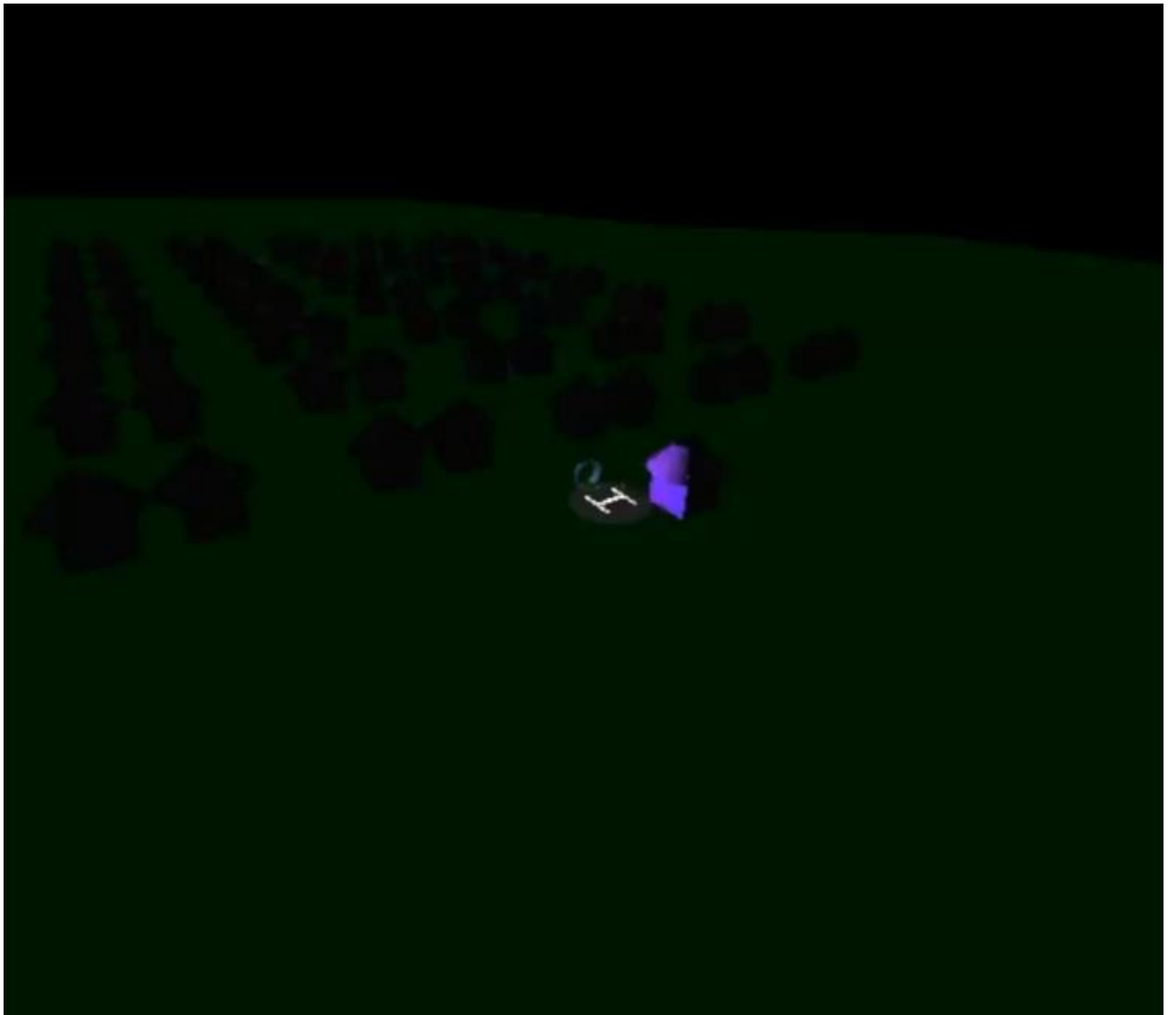
change to multiple colors by clicking the menu and choose the color.





Safari@ best desgin









Safari@ best desgin