Grafika komputerowa i komunikacja człowiek-komputer

Sprawozdanie z laboratorium

Data	Tytuł zajęć	Uczestnicy
10.11.2017 (realizacja zdalna)	OpenGL - interakcja z użytkownikiem	Iwo Bujkiewicz (226203)

Zadania

W ramach zadania należało napisać, na podstawie instrukcji laboratoryjnej, program realizujący trójwymiarowo wyświetlanie siatki czajnika Newella w rzucie perspektywicznym. Program miał umożliwić oglądanie czajnika z innego punktu, niż domyślny, a także obracanie czajnika i manipulację powiększeniem obrazu.

Kolejne etapy realizacji

W centrum zainteresowania była, jak zawsze, funkcja `render_scene()`. W odróżnieniu od poprzednich zadań, tym razem zawierała ona wywołanie funkcji `gluLookAt()`, w celu zdefiniowania położenia, zwrotu i obrotu kamery - coś, co nie było potrzebne w rzucie ortograficznym, ale stało się konieczne dla rzutu perspektywicznego.

```
void render_scene() {
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);

    glLoadIdentity();

    gluLookAt(camera[0], camera[1], camera[2], 0.0f, 0.0f, 0.0f, 0.0f, 1.0f, 0.0f);

    draw_axes();
```

Dalsza część funkcji `render_scene()` zawiera sposób obracania rysowanego czajnika Newella oraz sposób sterowania powiększeniem obrazu. Powiększenie zmieniane było poprzez modyfikację pionowego kąta rozwarcia ostrosłupa pola widzenia.

```
    if (action_state & 1<<0) {
        model_rotation[0] += mouse_pos_delta.x * degrees_per_pixel;
        model_rotation[1] += mouse_pos_delta.y * degrees_per_pixel;
    }
    if (action_state & 1<<1) {
        glMatrixMode(GL_PROJECTION);
        glLoadIdentity();

        fov_y += mouse_pos_delta.y * degrees_per_pixel;
        update_perspective();

        glMatrixMode(GL_MODELVIEW);
    }
    glRotatef(model_rotation[0], 0.0f, 1.0f, 0.0f);
    glRotatef(model_rotation[1], 1.0f, 0.0f, 0.0f);
```

Na sam koniec rysowany był oczywiście, białym kolorem, czajnik.

```
    glColor3f(1.0f, 1.0f, 1.0f);
    glutWireTeapot(3.0);

    glFlush();

    glutSwapBuffers();
}
```

Funkcja `resize_stage()` również została zmodyfikowana - zamiast operacji charakterystycznych dla rzutu ortograficznego, znalazł się tam kod aktualizujący widok perspektywiczny.

```
    aspect_ratio = (GLdouble)width/(GLdouble)height;
    update_perspective();
```

Użyta powyżej dwukrotnie funkcja `update_perspective()` służyła do aktualizacji widoku perspektywicznego z użyciem zmiennych globalnych `fov_y` oraz `aspect_ratio`.

```
void update_perspective() {
    gluPerspective(fov_y, aspect_ratio, 1.0, 30.0);
}
```

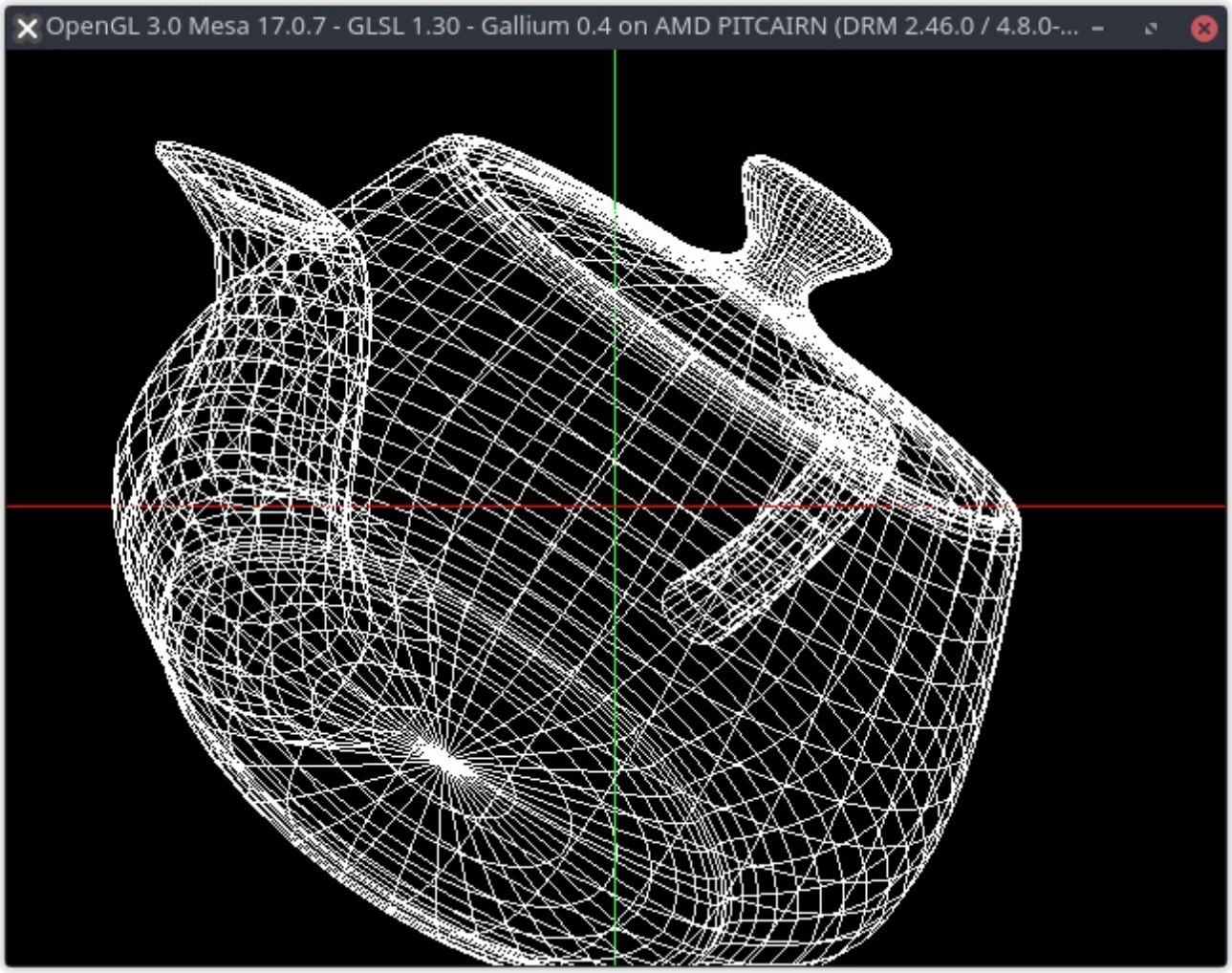
Aby zarejestrować zdarzenia naciśnięcia i zwolnienia przycisków myszy, skonstruowano funkcję `mouse_button_event()`.

```
void mouse_button_event(int button, int state, int x, int y) {
    if (button == GLUT_LEFT_BUTTON) {
        if (state == GLUT_DOWN) {
            last_mouse_pos.x = x;
            last_mouse_pos.y = y;
            action_state |= 1<<0;
        }
        else {
            action_state &= ~(1<<0);
        }
    }
    else if (button == GLUT_RIGHT_BUTTON) {
        if (state == GLUT_DOWN) {
            last_mouse_pos.x = x;
            last_mouse_pos.y = y;
            action_state |= 1<<1;
        }
        else {
            action_state &= ~(1<<1);
        }
    }
}
```

Ruch myszy wewnątrz okna programu, przy wciśniętym lewym lub prawym przycisku myszy, śledziła funkcja `mouse_motion_event()`.

```
void mouse_motion_event(int x, int y) {
    if (action_state) {
        mouse_pos_delta = (pixel){x - last_mouse_pos.x, y - last_mouse_pos.y};
        last_mouse_pos = (pixel){x, y};
        glutPostRedisplay();
    }
}
```

Rezultat działania programu prezentuje poniższy zrzut ekranu.



Kod źródłowy

Kompletny kod opisanych tu programów został załączony do sprawozdania w osobnych plikach.