Universitatea Tehnică „Gheorghe Asachi” din Iași

Facultatea de Automatică și Calculatoare

Domeniul: Calculatoare și Tehnologia Informației

Russian Checkers

Proiect la disciplina

Sisteme de prelucrare grafică

Student: Enachi Vasile

Coordonator: Paul Herghelegiu

I. Introducere

Proiectul de față și-a propus implementarea jocului “Russian checkers”, care reprezintă o variantă customizată a jocului clasic de dame.

S-a pornit de la un proiect existent, scris în limbajul C. Proiectul a fost reformatat, iar unele secvențe de cod au fost înlocuite cu funcții/structuri de date din librăria STL. De asemenea, s-a pus accent pe înlocuirea codului vechi de OpenGL, folosind Fixed Function Pipeline cu cod OpenGL folosind shadere. De asemenea au fost adaugate funcționalități noi, cum ar fi iluminare, texturare și normal mapping pe jocul existent.

Obiective atinse în cadrul proiectului

- compilarea lui în C++
- reformatarea codului (împărțirea codului în fișiere cu funcționalități diferite, înlocuirea structurii de date utilizate cu list din librăria STL, înlocuirea funcțiilor de căutare/filtrare cu funcții din librăria STL)
- adăugarea shader-elor
- înlocuirea codului vechi de OpenGL folosit pentru desenarea checkersboard-ului și a pieselor
- îmbunătățirea calității piesele de dame
- adăugarea iluminării
- adăugarea texturii pentru checkersboard
- adăugarea normal mapping pentru checkersboard
- adăugarea în UI a funcționalităților de modificare a poziției luminii, a puterii speculare a luminii, enable/disable textură/light/normal mapping
- adăugarea help-ului în UI
- modificarea funcționalităților existente a jocului (rotire 180 grade a tablei de board, rotire on/off pentru joc, salvare joc) și adaptarea lor la noile funcționalități adăugate

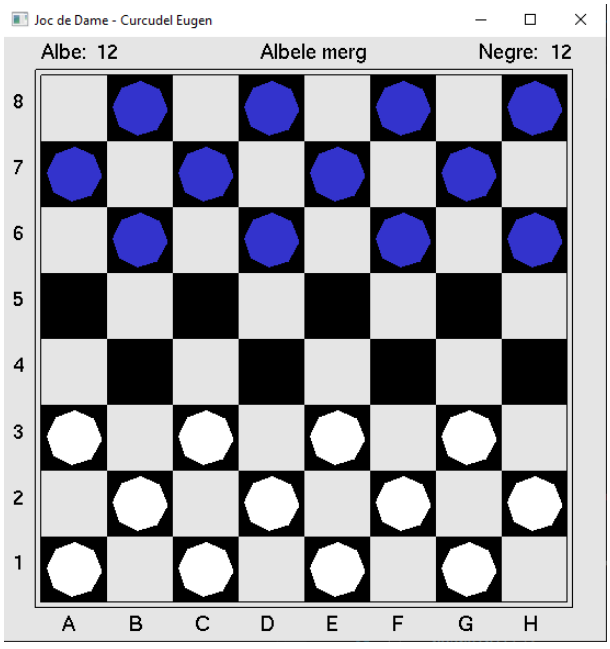


Figure 1. Captură din proiectul inițial

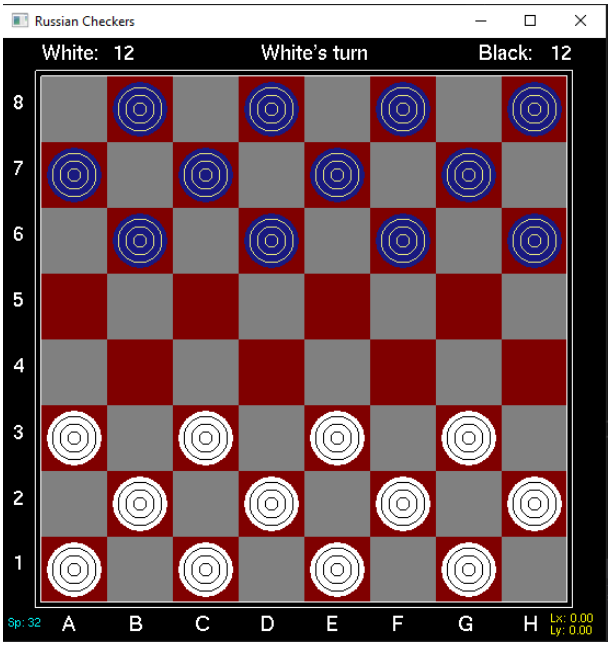


Figure 2. Captură proiect final (cu luminile, texturarea și normal mapping dezactivate)

II. Interfața cu utilizatorul

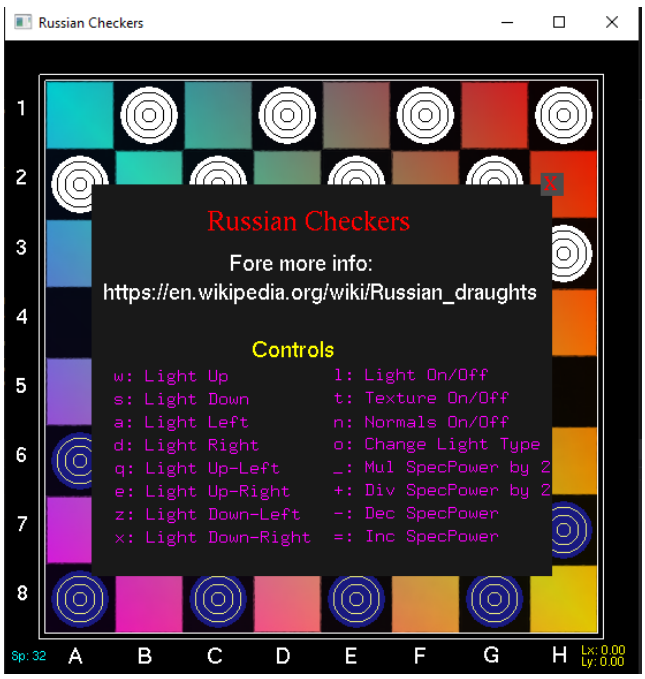


Figure 3. Help

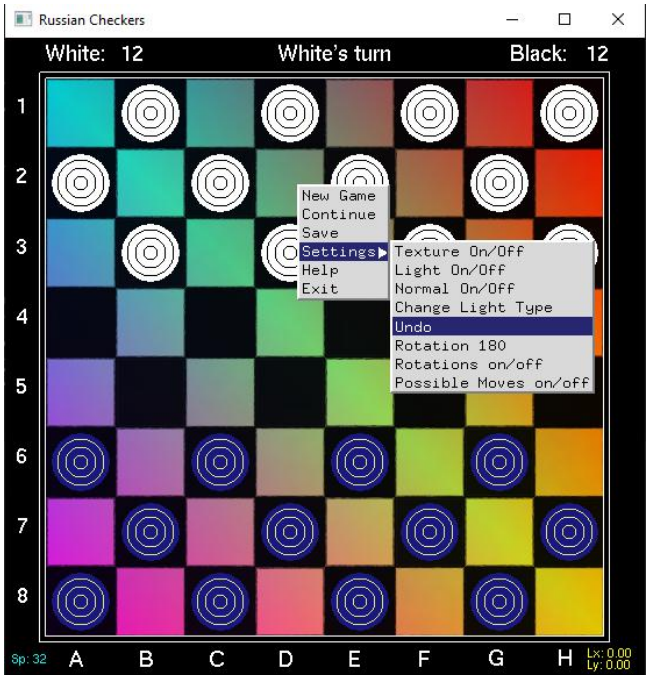


Figure 4. User-Interface options

Pentru a porni meniul de opțiuni se apasă click-dreapta. La acest apel, folosind funcțiile listener atașate aplicației, se determină poziția mouse-ului la momentul apăsării și se randează meniul la poziția specificată. Pentru adăugarea opțiunilor din joc s-a folosit funcția din utilitarul glut `glutAddMenuEntry` și `glutAddSubMenu`.

Opțiunile utilizatorului:

- New Game – Începerea unui joc nou
- Continue – Continuarea unui joc salvat (salvarea se face în fișierul "joc.check")
- Save – Salvează starea curentă a jocului (se salvează variabilele globale și tabloul board)
- Settings – Activarea submeniului de settings
  - Texture On/Off – Activarea/ dezactivarea texturii
  - Light On/Off – Activarea/ dezactivarea luminii
  - Normal On/Off – Activarea/ dezactivarea normal mapping-ului
  - Change light type – Schimbarea tipului luminii (theoretical type, experimental type)
  - Undo – Revino la starea curentă (se poate face doar un singur undo)
  - Rotation 180 – Rotația pieselor jucătorului pe părțile adverse
  - Rotations On/Off – Activarea/dezactivarea auto rotației la 180 grade după fiecare mișcare
  - Possible Moves On/Off – Activarea/ dezactivarea randării mișcărilor posibile
- Help – Pop-up a ferestrei de help
- Exit – Terminare program

III. Variabilele globale utilizate în program

```
//definirea constantelor
#define ROWS            8
#define COLUMNS         4

#define NO_CHECKER      0
#define BLACK_CHECKER   1
#define WHITE_CHECKER   2

#define CHECKER         0
#define KING            1

#define M_PI glm::pi<float>()

//uiManager
int MOUSEX = 0;
int MOUSEY = 0;
int PRESSED = 0;
int SIDE_COEF;

//definirea variabilelor globale
int WIN;

int TYPE1 = NO_CHECKER;
int TYPE2 = NO_CHECKER;
int ROTIRI = 0;
int GO = NO_CHECKER;
int JUMPED = 0;
int POS_MOVES = 0;
int HELP = 0;

std::list<std::pair<int, int>> jump_list;
std::list<std::pair<int, int>> check_list;
std::list<std::pair<int, int>> move_list;

//structura reprezentarii tablei de dame
struct square {
    float x;
    float y;
    unsigned int check : 2;
    bool type;
};
square board[ROWS][COLUMNS], undo_board[ROWS][COLUMNS];
```

```
std::pair<int, int> sel, to;

glm::mat4 projection_matrix, view_matrix, model_matrix;

glm::vec3 light_pos(a:0, b:0, c:5);
glm::vec3 view_pos(a:0, b:0, c:1);

glm::vec3 type1_color(a:1, b:1, c:1);   //white
glm::vec3 type1_selected_color(a:0.7, b:0.7, c:0.7);//gray

glm::vec3 type2_color(a:0.1, b:0.1, c:0.5); //blue
glm::vec3 type2_selected_color(a:0.2, b:0.2, c:0.8);//blue pale

glm::vec3 board_square_color(a:0.5, b:0.0, c:0.0);  //dark red

GLint enable_lighting = 0;
GLint enable_texture = 0;
GLint enable_normal = 0;
GLint light_type = 1;
GLfloat spec_power = 32;

GLuint lighting_shader_programme, texture_shader_programme;
GLuint board_texture, board_texture_normal;

float board_squares[ROWS][COLUMNS][12];
float full_board[32] = {
    // positions                        // colors           // texture coords
    -240.0f,    -240.0f,    0.0f,       1.0f, 0.0f, 1.0f,   0.0f, 1.0f,
     240.0f,    -240.0f,    0.0f,       1.0f, 1.0f, 0.0f,   1.0f, 1.0f,
     240.0f,     240.0f,    0.0f,       1.0f, 0.0f, 0.0f,   0.0f, 1.0f,
    -240.0f,     240.0f,    0.0f,       0.0f, 1.0f, 1.0f,   0.0f, 0.0f
};

float board_normalized[] = { -240.0f / 275, -240.0f / 275,
                             -240.0f / 275,  240.0f / 275,
                              240.0f / 275,  240.0f / 275,
                              240.0f / 275,  -240.0f / 275 };
```

Dintre variabilele globale, cele mai importante sunt:

- *board* – tablou de elemente de tip square care conține:
  - poziițiile pătratului pe table de dame (poziițiile sunt reprezentate în intervalul (-275.0, 275.0))
  - elemental ce se află în acel pătrat (nici un element, piesă albă, piesă neagră)
  - tipul piesei (piesă obișnuită, regină) (în program este verificată mai întâi dacă în pătrat este o piesă, apoi se verifică tipul acelei piese)
- *full_board* – tablou ce conține poziițiile (în intervalul (-275.0, 275.0)) pe tabla de dame unde se va aplica textura, culorile vărfurilor, poziițiile vârfurilor texturii
- *enable_lighting* – activarea/dezactivarea luminii
- *enable_texture* – activarea/dezactivarea texturii
- *enable_normal* – activarea/dezactivarea normalelor
- *spec_power* – puterea componentei speculare a luminii
- *jump_list* – listele de sărituri (ce trebuie efectuate de piese)
- *check_list* – listele de verificări cu mișcările posibile pe care le are o piesă
- *move_list* – listele de sărituri (ce trebuie efectuate de piese)
- *type1_color, type2_color* – culorile pieselor celor 2 jucători
- *type1_selected_color, type2_ selected_color* – culorile pieselor selectate a celor 2 jucători
- *light_pos* – definirea poziției luminii
- *sel, to* – varibile ce descriu poziția piesei selectate și poziția pătratului apăsat de user

IV. Logica rulării aplicației

În continuare se va analiza **main-ul** și funcțiile **timer** și **init**, care permit înțelegerea logicii de rulare al aplicației.

1. Main

```
int main(int argc, char** argv) {
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB | GLUT_DEPTH);
    glutInitWindowSize(width:550, height:550);
    glutInitWindowPosition(x:100, y:100);
    WIN = glutCreateWindow(title:"Russian Checkers");
    init_data();
    init();
    glutPassiveMotionFunc(passive_motion);
    glutDisplayFunc(display);
    glutMouseFunc(uimanager::mouse_listener);
    glutMotionFunc(uimanager::motion_listener);
    glutKeyboardFunc(uimanager::keyboard_listener);
    timer(s:0);
    glutMainLoop();
}
```

La pornirea aplicației se efectuează următorii pași definiți în main:

- se inițializează librăria Glut
- se creează un window de dimensiunea 550x550 setată începând de pe punctul (100, 100) de pe ecran
- se inițializează datele inițiale din program
- se atașează listenerii pentru keyboard și mouse precum și funcțiile de callback pentru display
- se apelează funcția timer în care are loc desfășurarea și logica întreg-ului program

2. Init

```
void init()
{
    // get version info
    const GLubyte* renderer = glGetString(name:GL_RENDERER); // get renderer string
    const GLubyte* version = glGetString(name:GL_VERSION); // version as a string
    printf(_format:"Renderer: %s\n", renderer);
    printf(_format:"OpenGL version supported %s\n", version);
    glDepthFunc(GL_ALWAYS);
    glEnable(cap:GL_DEPTH_TEST);
    glewInit();

    glGenBuffers(1, &vbo);
    glGenBuffers(1, &circle_vbo);
    glGenBuffers(1, &checkers_vbo);
    glGenBuffers(1, &crown_vbo);
    glGenBuffers(1, &board_texture_vbo);

    create_shader_program(vertex_shader_file:(char*)"shader/light_vertex.shader", fragment_shader_path:(char*)"shader/light_fragment.shader", [&]lighting_shader_programme);
    create_shader_program(vertex_shader_file:(char*)"shader/btext_vertex.shader", fragment_shader_path:(char*)"shader/btext_fragment.shader", [&]texture_shader_programme);

    set_texture((char*)"textures/board.jpg", [&]texture_shader_programme, [&]board_texture);
    set_texture((char*)"textures/board_normal.jpg", [&]texture_shader_programme, [&]board_texture_normal);

    create_menu();
    glClearColor(red:0.9f, green:0.9f, blue:0.9f, alpha:0.9f);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    glOrtho(left:-275.0, right:275.0, bottom:-275.0, top:275.0, zNear:0.0, zFar:1.0);
    board_init();
}
```

În funcția init se generează buffer-ele:

- vbo (pentru desenarea pătratelor de pe chessboard)
- circle_vbo (pentru desenarea cercurilor de pe piese)
- checkers_vbo (pentru desenarea pieselor)
- crown_vbo (pentru desenarea coroanelor de pe piesele regină)
- board_texture_vbo (pentru desenarea texturii) (conține și culori asignate)

Sunt create 2 programe shader (unul pentru texturi și normal mapping, altul obișnuit pentru iluminare) fiecare dintre ele conținând un vertex și un fragment shader.

Funcția create_shader_programme este responsabilă pentru crearea unui program shader, având ca parametri de intrare 2 path-uri pentru fișierele shader utilizate în programul shader respectiv. În funcția respectivă sunt verificate dacă fișierele shader sunt compilate corect, generându-se o eroare în caz contrar.

Funcția apelată create_menu, se folosește de funcțiile din utilitarul glut `glutAddMenuEntry` și `glutAddSubMenu` pentru a creea opțiunile din menu-ul jocului (activate tastând click dreapta).

Deoarece au fost păstrate randarea elementelor din afară checkersboard-ului (numerotare linii, coloane) și pentru a păstra numerotarea tablei de dame folosind doar întregi, a fost păstrată și funcția apelată aici: `glOrtho`, care mapează coordonatele implicite (-1.0, 1.0) cu coordonatele (-275.0, 275.0), adică cu dimensiunea ferestrei stabilite în main de 550x550.

În final, funcția board_init este folosită pentru inițializarea coordonatelor pătratelor tablei de dame (în coordonate din intervalul (-275.0, 275.0)), precum și inițializarea unor variabile globale.

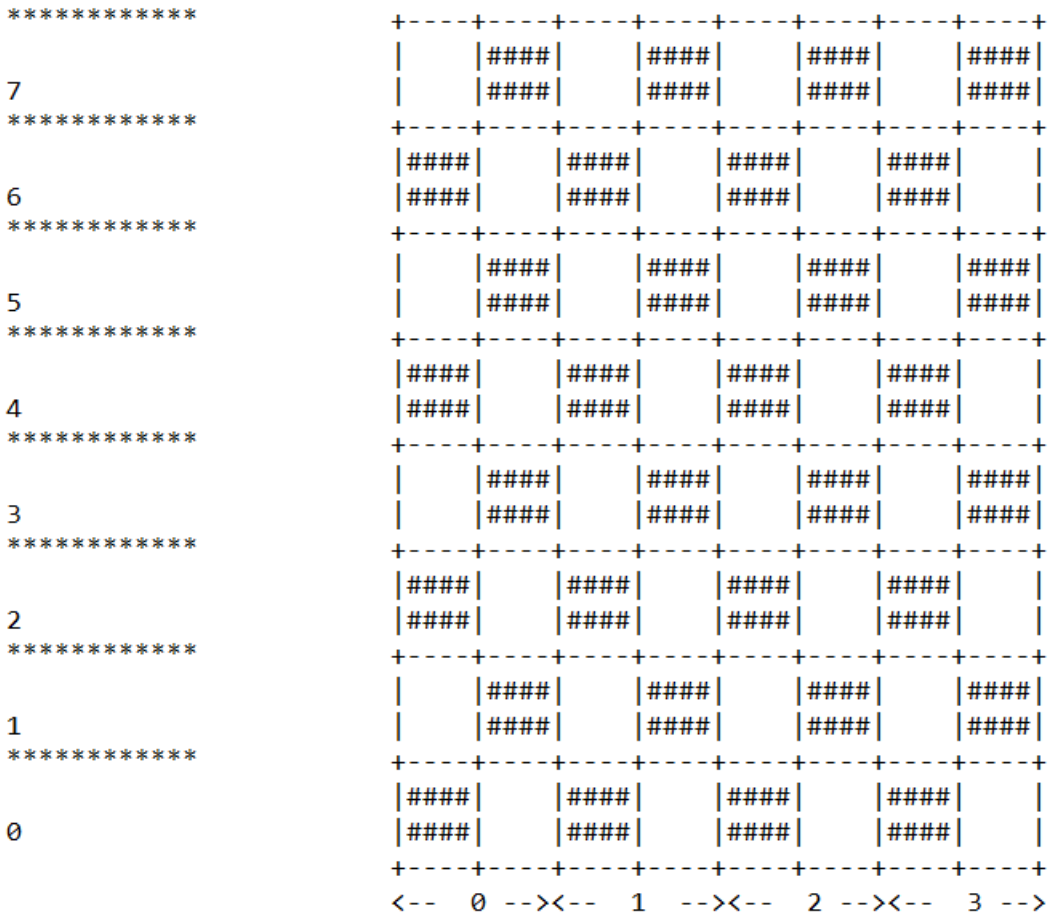
3. Timer

```
void timer(int s) {
    if (!check_list.empty()) check_list.clear();
    if (!jump_list.empty()) jump_list.clear();
    if (!move_list.empty()) move_list.clear();


    list_of_jumpes(&jump_list, &check_list);
    list_of_moves(move_list);

    display();

    for (auto i = 0; i < ROWS; i++)
        for (auto j = 0; j < COLUMNS; j++)
            if (MOUSEX < board[i][j].x + 30 && MOUSEX > board[i][j].x - 30 && MOUSEY < board[i][j].y + 30 && MOUSEY > board[i][j].y - 30)
                if (PRESSED) {
                    to.first = i;
                    to.second = j;
                }

    if (sel.first != -1 && to.first != -1) {
        copy_array(from: board, to: undo_board);
        if (move_is_legal(p: 1))
            put_checker();
    }


    //rechemarea recursiva a functiei timer()
    glutTimerFunc(time: 10, timer, value: 0);
}
```

Funcția time reprezintă “core-ul” programului, ea este apelată la fiecare 10 milisecunde prin intermediul funcției `glutTimerFunc`. Aici sunt eliberate cele 3 liste globale check_list, jump_list, move_list (vezi secțiunea **Variabile Globale**) și apoi actualizate în funcție de poziția curentă a pieselor pe tabla de dame prin intermediul funcțiilor list_of_jumpes și list_of_moves.

În funcția display are loc randarea tuturor elementelor de pe tabla de dame. Mai apoi este verificat dacă mouse-ul a fost apăsat de către utilizator și este verificat dacă click-ul mouse-ului reprezintă o mișcare validă a piese de dame (prin intermediul funcției move_is_legal), ulterior făcându-se această mișcare (prin intermediul funcției put_checker se actualizează tabloul global ce conține pozițiile pieselor).

V. Codificarea checkersboard-ului



Întrucât în jocul de dame pătratele utilizate sunt doar cele negre, în proiectul dat a fost economisit spațiul de memorie, declarând o matrice de 8x4 (în locul unei matrice de 8x8) pentru stocarea informațiilor despre pătratele jocului. Deși acest lucru ne permite utilizarea unui spațiu mic de memorie, el duce la o scădere a performanței jocului, întrucât în funcțiile de implementare a logicii jocului, pentru efectuarea unei mișcări/salt sau detectarea mișcărilor posibile este necesară identificarea coordonatelor fiecărui pătrat. Întrucât cele 8 coloane de pătrate sunt codificate ca 4 coloane, este nevoie pentru fiecare identificare a unui pătrat folosirea divizibilității cu 2 a rândurilor.

VI. Iluminarea

Modelul de iluminare folosit este modelul Phong (iluminarea este determinată de componetele ambientală, specular și difuză).

Fenomenul de iluminare este implementat atât în fragment-shader obișnuit (în lipsa texturii), cât și în fragment-shader-ul destinat randării texturilor cu normal mapping. În ambele cazuri, către shadere-le respective sunt transmise prin intermediul variabilelor uniforme:

- poziția luminii
- poziția viewer-ului
- culoarea obiectului de randat
- variabila de tip bool pentru dezactivarea/activarea luminii
- tipul luminii (experimental/teoretic)
- exponentul componenetei speculare

În cadrul aplicației, exponentul componentei speculare a fost limitat la o mărime între [1, 1024].

Poziția luminii poate fi schimbată folosind butoanele **w**, **a**, **s** ,**d** ,**q** ,**e** ,**z** ,**x**. Pentru activarea/dezactivarea luminilor se apasă tasta l, iar pentru schimbarea tipului luminii se apasă tasta o.

Pentru modificarea exponentului puterii speculare se folosesc tastele -, _ (micșorare), =, + (mărire).

În cadarul aplicației au fost adăugate 2 tipuri de lumini, cel teoretic și cel experimental. Cel teoretic respectă formula modelului Phong descris în teorie. Cel experimental amplifică componenta ambientală cu lumina sursei de lumină (la fel ca componenta difuză).

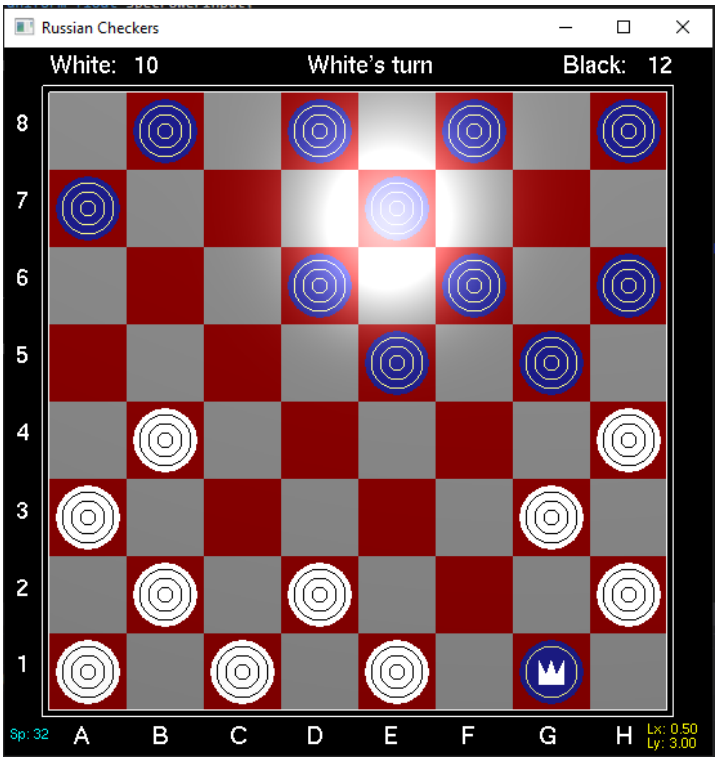


Figure 5. Tipul luminii – teoretică, cu exponentul puterii speculare de 32

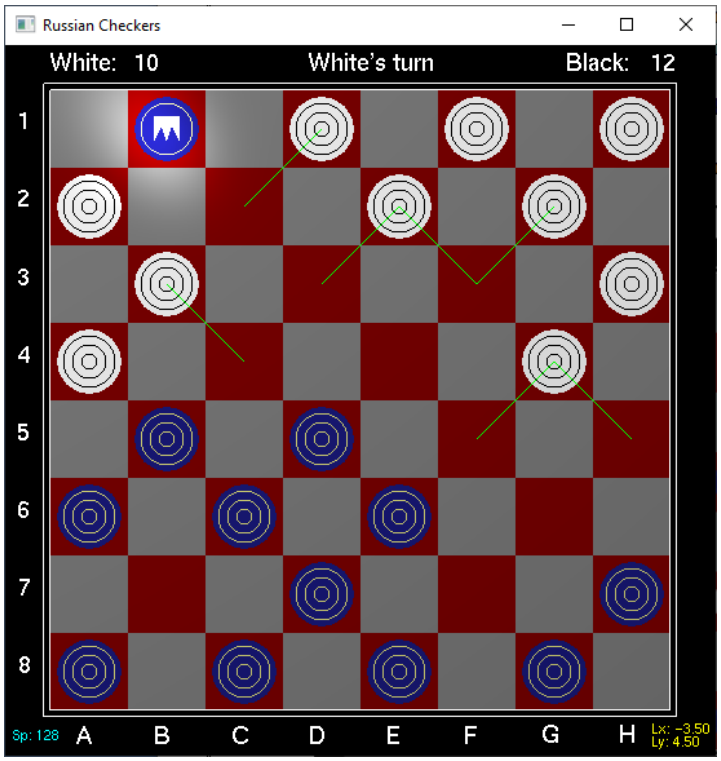


Figure 6. Tipul luminii – experimental, cu puterea specular de 128

VII. Texturarea și normal mapping

Pentru citirea texturii și a imaginii normalelor din fișier este folosită librăria stb_image. Componentele RGB din imaginea normalelor poate fi percepută ca fiind coordonatele noilor direcții ale normalelor.

Randarea texturii este determinată de variabila globală enable_texture, care poate fi activată/dezactivată de către utilizator prin apăsarea tastei **t**. De asemenea, activarea/dezactivarea normal-mapping-ului se face prin apăsarea tastei **n**.

Pentru o utilizare eficientă a texturării, în funcția definită set_texture s-au atribuit valorile corespunzătoare proprietăţilor texturii descrise de constantele GL_TEXTURE_MIN_FILTER, GL_TEXTURE_MAX_FILTER, GL_TEXTURE_WRAP_S, GL_TEXTURE_WRAP_T.

În variabila globală full_board este conținut tabloul de valori ce reprezintă:

- pozițiile pe tabla de dame unde să fie aplicată textura
- culoarea marginilor
- pozițiile texturii încărcate ce vor fi aplicate pe tabla de dame

Textura și normalele sunt transmise către fragment_shader-ul pentru texturi prin intermediul variabilelor uniforme.

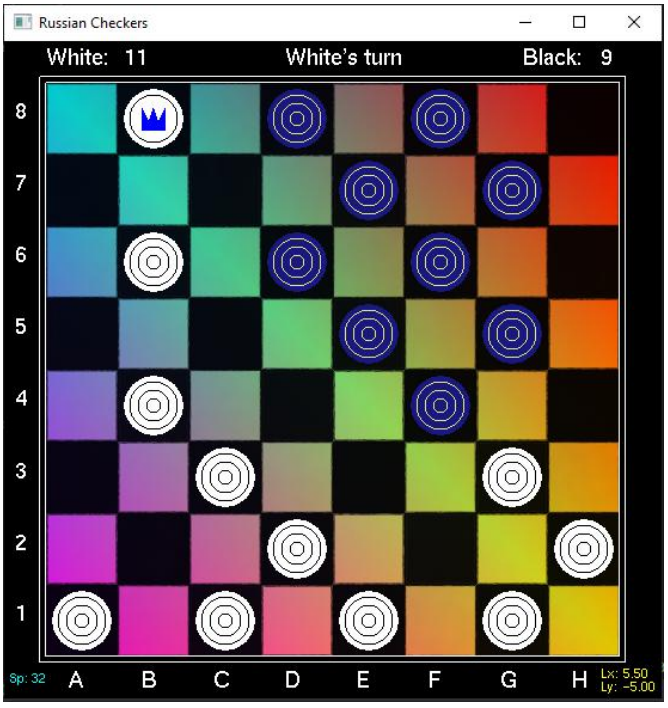


Figure 7. Texture enable, light off

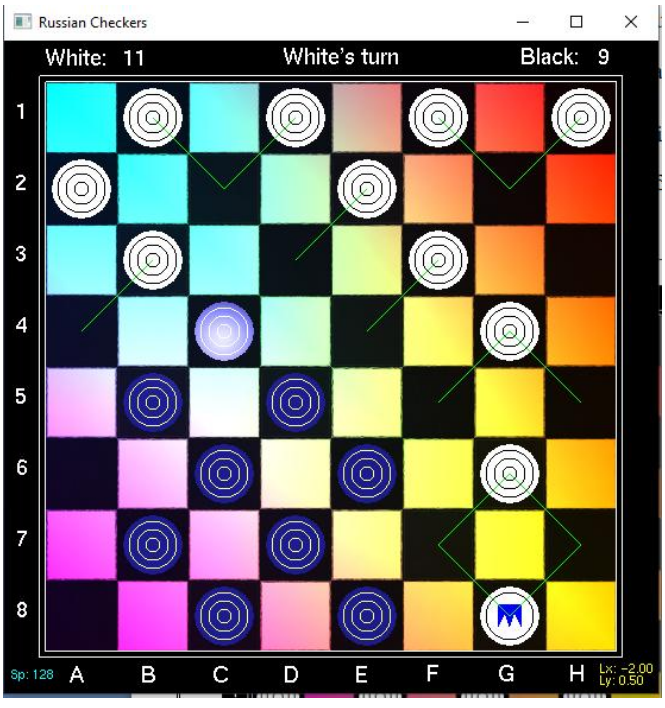


Figure 8. Texture enable, light on, specular exponent – 128, possible moves enabled, normal mapping - off

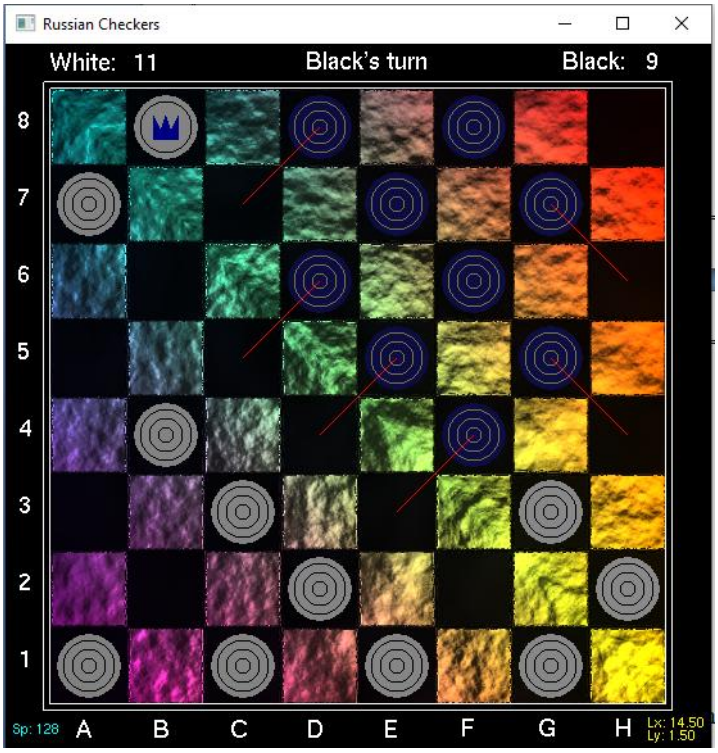


Figure 9. Texture enable, light on, specular exponent – 128, possible moves enabled, normal mapping - on

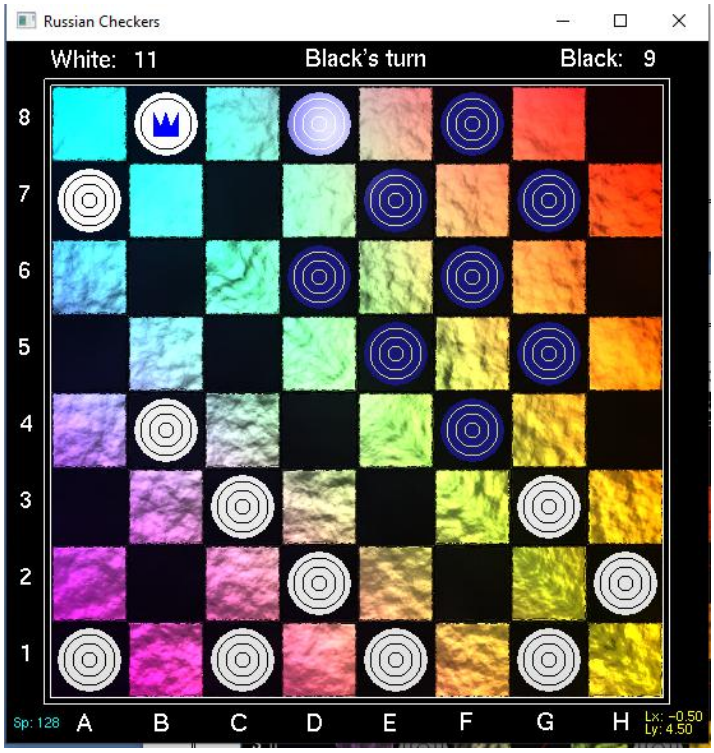


Figure 10. Texture enable, light on, specular exponent – 128, possible moves disabled, normal mapping - on

VIII. Funcția de display

Funcția display este cea care este apelată în funcție timer. Aici sunt apelate toate funcțiile de afișare a componentelor jocului:

- background
- checkersboard
- piese simple
- piese regină
- mișcări posibile
- exponentul componentei speculare (stânga jos)
- poziția sursei de lumină (dreapta jos)

```
727  //functia de desenare grafica principala
728  void display() {
729      //curatam ecranul
730      glClear( mask: GL_COLOR_BUFFER_BIT);
731      glUseProgram(0);
732      draw_background();
733      draw_water();
734
735      glm::vec3 color;
736
737      if (enable_texture)
738          draw_board_with_texture(board_texture_vbo);
739      else
740          for (auto i = 0; i < ROWS; i++)
741              for (auto j = 0; j < COLUMNS; j++)
742                  draw_board_square(i, j, board_square_color, [&]vbo);
743
744      //draw checkers pieces
745      for (auto i = 0; i < ROWS; i++) { ... }
818
819      //desenam cifrele din stinga tablei, literele de jos si doua linii de contur
820      draw_around(SIDE_COEF);
821      draw_light_position(light_pos.x, light_pos.y, SIDE_COEF);
822      draw_light_spec_power(spec_power, SIDE_COEF);
823
824      //Desenam toate miscarile posibile daca exista
825      if (POS_MOVES) draw_possible_moves();
826
827      //desenam output-ul meniului "Ajutor"
828      if (HELP) { ... }
837      else { ... }
867      glFlush();
868  }
869
```

IX. Concluzii și observații

La programul dat poate fi adăgată funcționalitatea de resize, adică funcțiile de afișare trebuie reformatate să depindă de înălțimea și lățimea ferestrei (în programul respectiv afișarea se face folosind dimensiunea ferestrei de 550x550).

Programul dat poate fi eficientizat prin schimbarea metodei de indexare a tabelei de dame (folosind un tablou 8x8 în schimbul celui existent de 8x4, care necesită multe verificări pentru a accesa un pătrat respectiv).

De asemenea, aplicația dată poate fi extinsă pentru a juca împotriva unul calculator, folosind elemente de inteligență artificială.

Referințe

- Russian checkers
- Learning OpenGL
- DocsGL Documentation
- Cherno's OpenGL Series