Automatică și calculatoare

**Anul III**

**SIMULATOR DE FIZICĂ:**

**SIMULAREA ÎN TIMP REAL  
A NISIPLUI CĂZĂTOR**

**Candidat: Mironică Vasile**

**Coordonator științific: Lect. Norbert Gal-Nădășan**

Sesiunea: Iunie 2024

**1.0 Introducere**

Din cauza limitărilor tehnologice și a dorinței de a imita ”graficele vieții reale”, jocurile video tind să arate o parte foarte mică a lumii în care jucatorul se află. Acest fapt conduce direct la lipsa dorinței de a juca un joc a doua oară – deoarece jucătorul poate explora, cu *relativ* puțin efort, multitudinea de acțiuni preprogramate, precum și multitudinea de obiecte create de artiști, lăsând puțin spațiu pentru imaginație.

Dacă se face un compromis între **realism** și **puterea de procesare**, se poate rezolva problema unei lumi statice, astfel aducându-se în joc o lume în care acțiunile jucătorului influențează direct întreaga lume în același timp, conducând la mecanici foarte interesante, consecințe neprevăzute și comportament imprevizibil. Ideile acestui proiect sunt foarte simple:

1. Tratează toate elementele din lume unul câte unul;
2. Elementele pot interacționa doar cu elementele din imediata lor împrejurime.

Prima idee conduce direct la ce vrem să obținem – o lume cât se poate de dinamică în care interacțiunile nu sunt ”optimizate în neexistență”, iar a doua idee este prezentă ca o limitare a primei idei, ca să economisim putere de procesare, permițând jocului să ruleze în timp real. O a treia idee a proiectului, care nu întotdeauna se menționează, dar are afect *masiv* asupra experienței, este **performanța** – jocul trebuie să se simtă fluid în mișcare, jocul trebuie să răspundă ”instant” la acțiunile jucătorului (se va reveni la această idee în capitolul Performanță).

Limbajul de programare predominant în lucrare este **C++** - un limbaj foarte puternic și bine cunoscut în comunitate pentru stabilitatea, flexibilitatea și performanța sa, dovedite pe parcursul întregii sale vieți de nenumărate ori. Partea grafică(afișarea la ecran) este adusă de către librăria open-source **SFML** – **S**afe and **F**ast **M**ultimedia **L**ibrary, care vine la pachet și cu funcții pentru citirea intrărilor de la mouse și tastatură. Afișarea la ecran este accelerată pe placa video cu ajutorul proiectului **OpenCL** de la organizația Khronos.

**2.0 Prezentare generală**

Ideea se comportă, la bază, ca un Automat Celular – se începe cu starea zero, iar starea următoare este determinată calculand rezultatul aplicării unor reguli asupra elementelor din starea precedentă. Simulatorul este capabil să simuleze 8 substanțe: aer, nisip, piatră, apă, ulei, abur, foc și fum. Regulile comportamentului elementelor se pot clasifica în două tipuri:

1. **Reguli simple** – schimbarea poziției elementului este singura consecință a trecerii de la o stare la alta.
2. **Reguli compuse** – în urma interacțiunilor cu un alt element, acesta își schimbă forma.

Un element poate avea multiple reguli simple, precum și multiple reguli compuse.

**2.1 Piatră**

Elementele de piatră servesc ca platformă pentru alte elemente. Acestea sunt complet staționare – nu sunt mișcate de nimic și ele însele nu sunt afectate de gravitație.

**2.2 Nisip**

Un element de nisip trebuie, în primul rând, să respecte gravitația.

* **Regula 1:** Dacă o celulă de nisip nu are nimic direct sub ea, aceasta ar trebui să înlocuiască elementul direct sub ea (fig. 2.1).

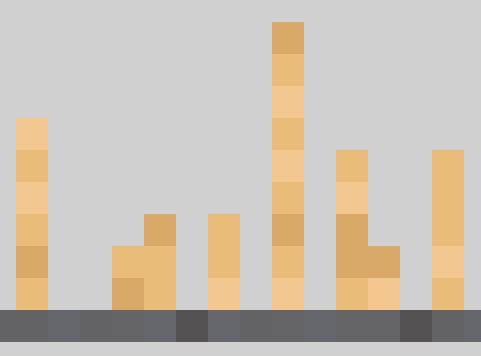


Figura 2.2: Grămadă de nisip, folosind doar regula 1



Figura 2.1: Nisip, regula 1

Însă, doar această regulă generează comportament nenatural, nisipul formând ”piloni”, când nisipul tinde să se ”imprăștie” în viața reală. Această problemă poate fi ușor rezolvată introducând o regulă nouă:

* **Regula 2:** Daca o celulă de nisip nu are nimic direct în stânga-jos (sau în dreapta-jos), celula de nisip trebuie sa inlocuiască respectiva celulă (fig. 2.2).
  1. Dacă ambele poziții sunt valabile, se va selecta una aleatoriu cu șanse egală.
  2. Dacă nicio poziție nu este valabilă, celula de nisip stă pe loc.

Figura 2.3: Nisip, regula 2

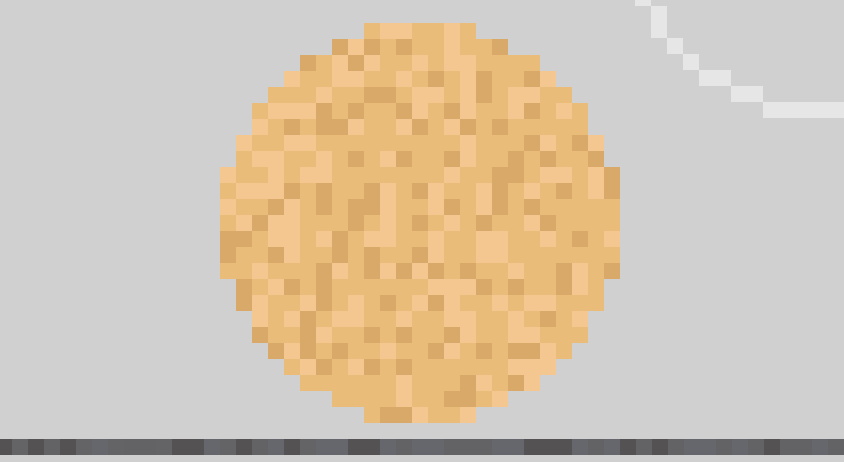
Figura 2.4: Grămadă de nisip în stare inițială

Figura 2.5: Grămadă de nisip simulată conform regulilor 1 și 2

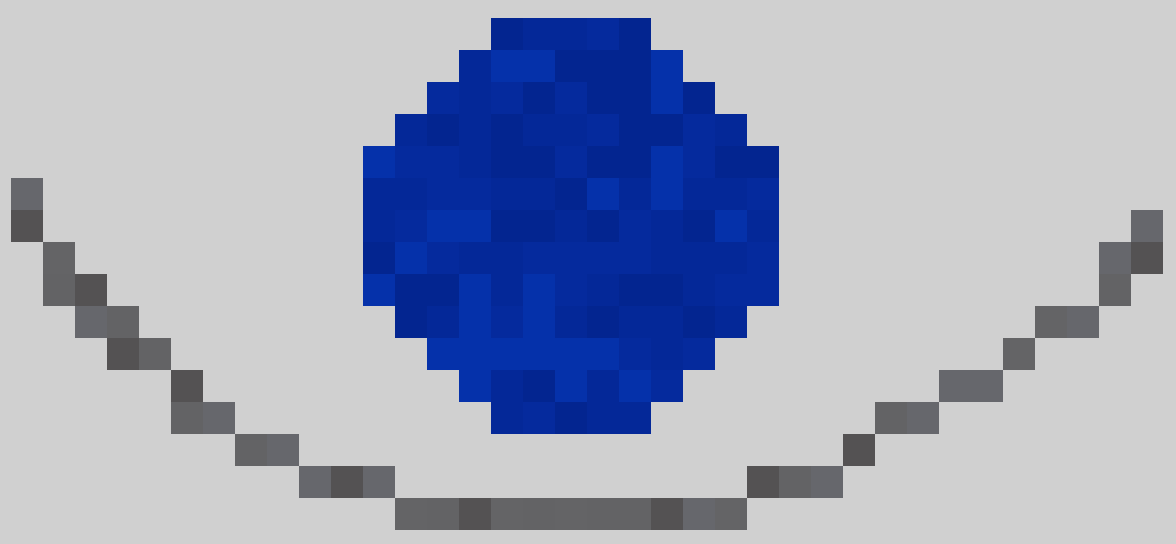
**2.3 Apă**

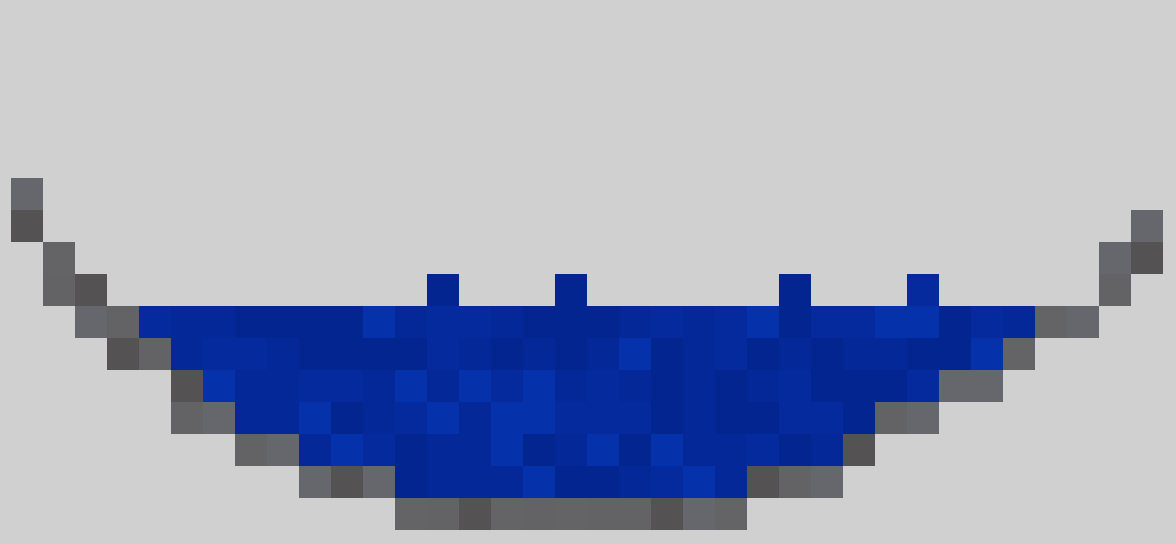
Apa este fluidă, deci,în contextul simulării, are proprietatea de a umple spațiul în care se află. Deasemenea, apa este afectată de gravitație și are proprietatea de a se împrăștia – rezultă ca putem folosi aceleași reguli ca și la nisip, adaugând doar una nouă:

* **Regula 3:** Dacă o celula de apă nu are un alt element direct în stânga (sau în dreapta) ei, celula de apă trebuie sa inlocuiască respectiva celulă.
  1. Dacă ambele celule sunt valabile, se va selecta una aleatoriu cu șanse egale.
  2. Dacă regula a fost aplicată deja, trebuie să fie aplicată din nou în aceeași direcție.
  3. Dacă nicio poziție nu este valabilă, celula de apă stă pe loc.

**Observație:** Subregula **(b)** este adăugată, deoarece în lipsa ei celula de apă va ajunge mereu în aceeași poziție (asumând că durata simulării este infinită), din cauza faptului că șansa de a merge în dreapta sau în stânga este egală.

Figura 2.6: Apă, regula 3

Figura 2.7: Apă în starea inițială

Figura 2.8: Apa după ce a umplut volumul dat

**2.4 Fum**

Fumul este un gaz, acesta are aceleași proprietăți ca și un fluid – însă inversate pe axa verticală, rezultă că i se aplică aceleași reguli ca și apei, însă calculele coordonatelor sunt inversate pe axa verticală.