L-systémy v počítačovej grafike

Patrik Sakáč

Vedúci práce: RNDr. Rastislav Krivoš-Belluš, PhD

Ústav Informatiky, Prírodovedecká fakulta, UPJŠ Košice

Abstrakt

Práca sa zaoberá rastom stromov a kríkov v počítačových hrách pomocou L-systémov. V hrách je rast stromov, kríkov a iných rastlín bežným problémom, s ktorým sa vývojári stretávajú. Pri veľkom počte inštancií sa upúšťa od tejto funkcionality, pretože si vyžaduje veľké nároky na operačnú pamäť, procesor či grafickú kartu. Inou možnosťou je znižovanie nárokov na konečný grafický vzhľad. Naše riešenie je vytvorenie obrázka pomocou L-systémov, ktorý bude uchovávať dôležité informácie o raste daného druhu. Tento obrázok následne použijeme a implementujeme rast.

L-systémy

Zaoberali sme sa s generovaním reťazcov pomocou gramatík popisujúcich L-systémy. Poznáme parametrické a bezparametrické pravidlá. Pri parametrických pravidlách, po nahradení každého pravidla musíme ešte dosadiť aktuálne hodnoty parametrov do premenných. Následne vyhodnotíme výrazy v parametroch a máme nové aktuálne hodnoty parametrov. Pri bezparametrických pravidlách len nahradzujeme jednotlivé pravidlá.

Príklad bezparametrickej gramatiky:

$\Sigma=\{0,1,[,]\}$

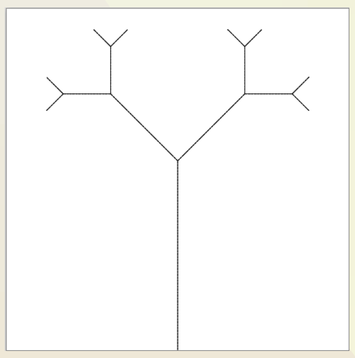
$S=0$

$P=\{(1\rightarrow 11), (0\rightarrow 1[0]0)\}$

1. iterácia: 1[0]0

2. iterácia: 11[1[0]0]1[0]0

3. iterácia: 1111[11[1[0]0]1[0]0]11[1[0]0]1[0]0



Obrázok 1. Vykreslená tretia iterácia.

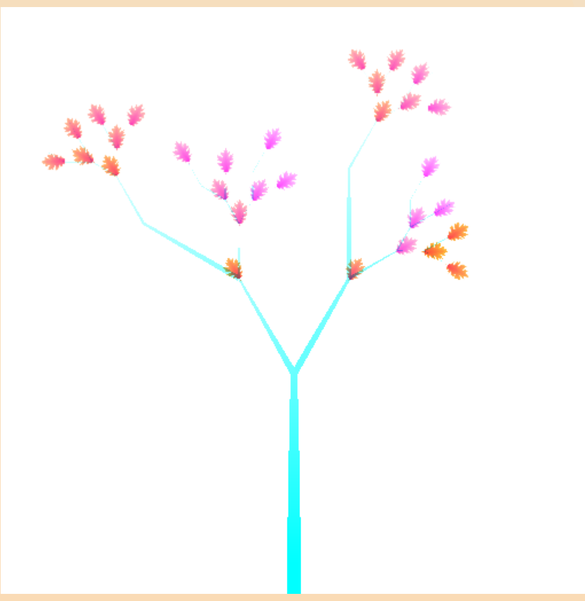
Ciele

1. Analyzovať existujúce riešenia a aplikácie Lindenmayerových systémov (L-systémov) v počítačovej grafike a počítačových hrách.
2. Implementovať viaceré typy L-systémov (napr. kontextové, parametrické) pri tvorbe 3D-objektov.
3. Analyzovať efektívnosť implementácie pri generovaní komplexnej scény.

Maska

Algoritmus dostane na vstupe reťazec, ktorý reprezentuje nejaký obrázok. Tento reťazec je vytvorený pomocou L-Systémov. Reťazec môže byť parametrický alebo bezparametrický. Pri reprezentácii znakov v reťazci sa používa tzv. korytnačia grafika. Korytnačka začína zo svojej štartovacej pozície a so štartovacou rotáciou. Postupným čítaním reťazca po jednotlivých znakoch korytnačka vie akú operáciu má vykonať.

Na obrázku č. 2 môžeme vidieť obrazec, ktorý popisuje stav v každom momente rastu konkrétneho konára stromu. Pri vytvorení masky bol použitý obrázok č. 3, ktorý popisuje stav jednotlivého listu v každej fáze rastu.



Obrázok 2. Rast konára.

Obrázok 3. Rast listu.

Literatúra

Pedrosa D.S.: GALSYS - Procedural Creation of Trees - through Combination of Genetic Algorithms and Lindenmayer Systems, 2014, ISBN 978-3639643763

Quigley E., et al.: Real-time Interactive Tree Animation, IEEE Transactions on Visualization and Computer Graphics Vol 22, 2017

Fornander P.: Game Mechanics Integrated with a Lindenmayer System, thesis, School of Computing, Blekinge Institute of Technology, Sweden, 2013

Hornby G.S., Pollack J.B.: Evolving L-systems to generate virtual creatures, Computer & Graphics (25) 6, 2001, pp. 1041-1048

Trube B.: Fractals - A Programmer's Approach, Amazon Digital Services, 2013

Prusinkiwicz P., Hanan J.: Lindenmayer systems, fractals, and plants, Springer Science & Business Media, Springer Verlag, 1989, ISBN 978-0387970929

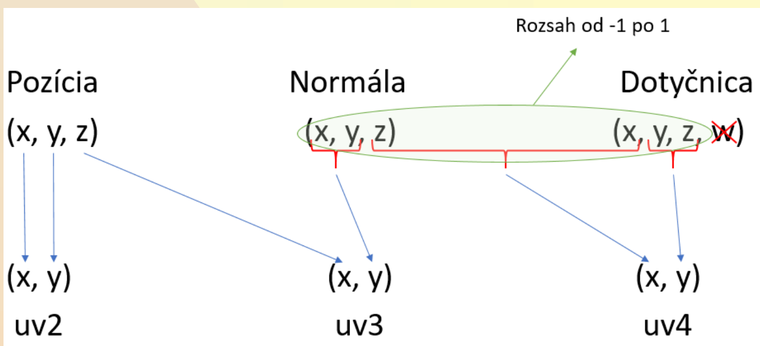
www.PosterPresentations.com

Mesh blending

Animácia pomocou prelínania jedného meshu na iný sa zvykne nazývať "*mesh blending*" alebo aj "*morph target*" či "*blend shapes*". Takáto animácia sa najčastejšie používa vo filmoch a hrách na animácie tvárí osôb. Pri veľmi detailných objektoch, ktoré sú tvorené miliónmi vertexov je to zvyčajná metóda. Dá sa tak veľmi jednoducho simulovať rozprávanie ľudí. Stačí mať tvár vytvorenú v dôležitých mimických pozíciách.

My použijeme takúto animáciu na rast stromov. Strom v jednej fáze sa postupne zmení na strom v nasledujúcej fáze rastu. Unity však nepodporuje takýto typ animácie a preto si musíme vytvoriť náhradu. Náhrada spočíva vo vytvorení objektu, ktorý obsahuje informácie potrebné na takúto animáciu. My sme si zvolili objekt, ktorý uchováva v sebe n-1 meshov, ak n je počet meshov, čo sa má animovať. Každý i-ty mesh si uchováva potrebné informácie o nasledujúcom (i+1) meshi. Potrebné informácie sú pozície, normály a dotyčnice vertexov.

Problémom je, kam si tieto potrebné údaje zapíšeme. Existujú dve možnosti kam ich môžeme zapísať. Prvou možnosť je zapísať potrebné údaje do obrázka. Druhou možnosťou je komprimovať údaje a použiť voľné premenné v meshi Za voľné premenné sa považujú také, ktoré nevyužívame pri žiadnej aktivite vrátane samotného renderovania. Keďže sme v 3D priestore tak pozícia, normála a dotyčnica obsahujú každá po tri premenné. Voľné premenné máme "*uv2*","*uv3*","*uv4*". Tieto obsahujú po dve premenné. Potrebujeme teda zapísať deväť premenných do šiestich. Využijeme fakt, že normála a dotyčnica majú rozsah od -1 po 1. Takže hodnoty normály a dotyčnice skomprimujeme po dvoch do jednej premennej. Neskôr, ak vieme ako sa dostať k jednotlivým hodnotám, tak môžeme urobiť lineárnu interpoláciu. Tú použijeme medzi dvoma pozíciami vertexov, ich normálami aj dotyčnicami.



Obrázok 5. Rozdelenie premenných.

Obrázok 4. Finálna animácia.

Zhrnutie

Implementovali sme rast stromu, ktorý je animovaný na grafickej karte. Použili sme vývojové prostredie Unity. V Unity sa dajú veľmi efektívne vykresľovať statické objekty. Pri implementácii sme použili generovanie L-systémov a známu animáciu mesh blending, ktorá nie je súčasťou vývojového prostredia Unity. Rast sme vyskúšali na veľkom počte inštancií a zistili sme, že je takmer tak efektívny ako statické objekty. K rastu sme pridali aj animáciu vetra.

Výsledky analýzy scén

	Static scene	Grow trees scene
Render thread	3.5ms	4.3ms
Graphics	43.4ms	45.4ms
SetPass calls	9	6
Batches	33	33
Saved by batching	14 188	14 188