SPRACOVANIE TÉM VÝTVARNEJ INFORMATIKY V PROGRAME CINEMA 4D

MARTIN GRMAN

PV097

JESEN 2012

Obsah

Úvod

- Tento projekt slúži na ukázanie a vysvetlenie niektorých pokročilých funkcii Cinemy 4D a ich použitie vo výtvarnej informatike.
- Cely tutoriál je rozdelený na niekoľko časti podľa témy. Plus úvodné predstavenie pokročilých funkcii.
- Cinema 4D je 3D modelovací program od spoločnosti Maxon.
- Je to rozsiahli nastroj umožňujúci pracovať od 3D modelov cez animáciami až po fyzikálne simulácie.
- Tento prehľad je vytváraný na anglickú verziu R13 Studio, je použiteľný z veľkej časti aj v iných verziách.
- Jednotlivé predstavené modeli budú vytvárané s úmyslom ďalšieho spracovania do výtvarného diela.
- Predpokladá sa znalosť Cinemy 4D v rozsahu predmetu Základy počítačové grafiky

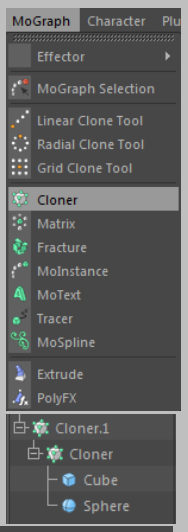
Predstavenie pokročilých funkcii

- V tejto kapitole sa nachádza úvod do funkcii *MoGraph, XPresso, Thinking Particles*.
- Preto je potrebná Cinema 4D R13 verzia Studio ktorá všetky tieto funkcie obsahuje. Ine verzie ako Prime, Brodcast či Visualize obsahujú len niektoré.

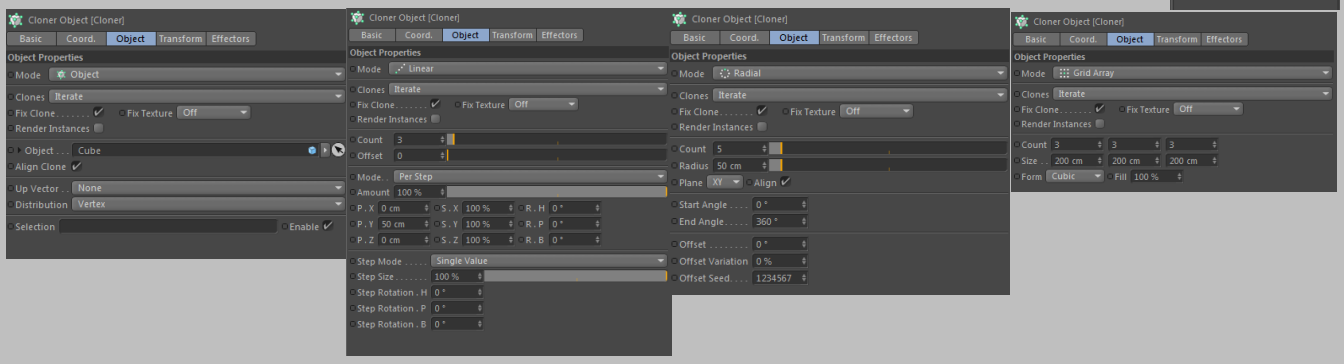
MoGraph

- *MoGraph* je súbor nástrojov, ktoré slúžia napríklad na :
    - Klonovanie objektov
    - Ich ovplyvňovanie pomocou efektorov
    - Prácu s textom a krivkami
    - Ako nové Deformery
- Nebudem predstavovať všetky možnosti *MoGraph* ale len tie ktoré budeme využívať
- Odporúčam si pripadne príkazy z *MoGraph,* ktoré tu nebudeme preberať naštudovať v *Help*. Sú tam ukážky a je celkovo vynikajúci, keď niečo nie je jasné
    - Pravým tlačidlom kliknúť na akýkoľvek príkaz alebo objekt a vybrať z ponuky *Show help*.
- Konkrétne ukážky použitia budú na konci kapitol
- Nasleduje hlbšie vysvetlenie základov *MoGraph*

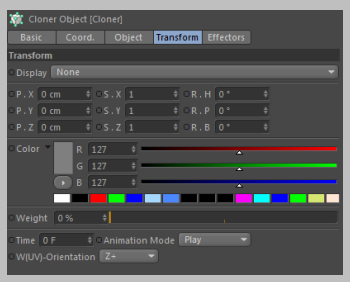
Cloner Object



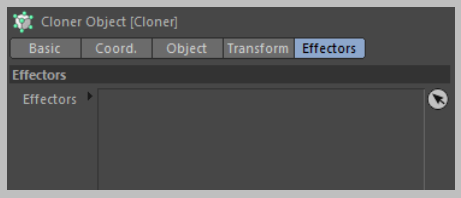
- Tento objekt slúži na parametrické klonovanie objektov
- Nachádza sa v menu *MoGraph*
- To znamená že objekty sa klonujú ale v hierarchii je to zobrazene ako jeden objekt, kde môžeme kedykoľvek meniť parametre klonovania
- Môže ako zdroj využiť aj sám seba, čím môžu vznikať veľmi zaujímavé útvary
- Objekty sa ukladajú pod neho v hierarchii
- Objekty môžeme klonovať lineárne, radiálne, do mriežky alebo podľa objektu
- Nastavenia *Cloner* spočívajú v troch špecializovaných taboch



- *Tab Object*
  - *Mode :* tu vyberáte mód klonovania, podľa neho sa zobrazia parametre
  - *Clones :* tu nastavíte ako sa vyberajú objekty zo všetkých čo sú pod nim v hierarchii
  - *Count :* tu nastavíte počet klonov na objekte
  - *Object-Distribution :* tu nastavíte ako sa distribuujú klony na povrch či objem cieľového objektu
  - *Linear-Mode :* tu nastavíte mód zmeny, či nastavíte konečnu zmenu a interpoluje alebo zmenu za klon
  - *Linear-Step * :*rôzne nastavenia tykajúce sa zmeny, v prípade potreby konzultujte help
  - *Radial-Radius :* polomer v akom sa bude klonovať
  - *Radial-Plane :*rovina v ktorej sa klonuje radiálne
  - *Grid-Count :* počet klonov v osách x,y,z mriežky
  - *Grid-Size :* celková veľkosť mriežky
- *Tab Transform*
  - Transformácia tu zadaná sa aplikuje na objekt ešte pred klonovaním
  - Môžeme zmeniť pozíciu, veľkosť či rotáciu objektu
  - Taktiež môžeme vybrať farbu akou sa budú zobrazovať klony

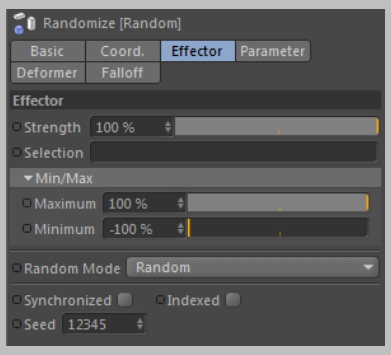


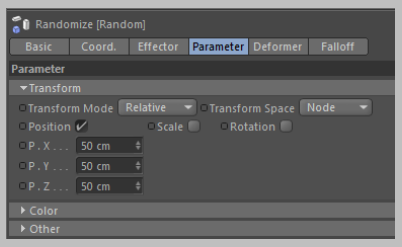
- *Tab Effectors*
  - Tu sa nachádza zoznam všetkých efektorov ktoré ovplivnujú daný *Cloner*

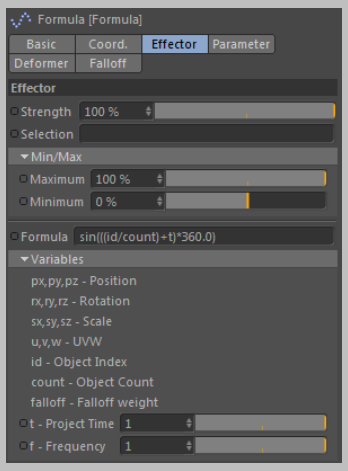


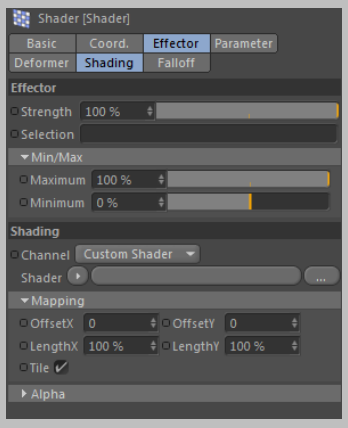
Effektory

- Nachádzajú sa v menu *MoGraph* a podmenu *Effector*
- Tieto objekty upravujú chovanie *Cloner*
- Tak že podľa funkcie, shaderu, objemu menia polohu, veľkosť a rotáciu objektu
- Po vytvorený musí byt priradený do *Clonera* do časti *Effectors*
- Ak bol označený *Cloner* tak novo vytvorený *Effector* bude automaticky k nemu priradený
- Je ich veľa druhov s rôznou funkcionalitou
- Najpoužívanejšie sú *: Random, Formula, Shader, Step*



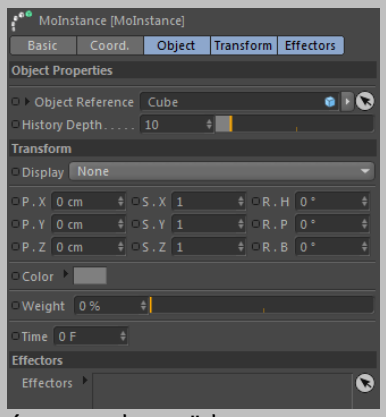






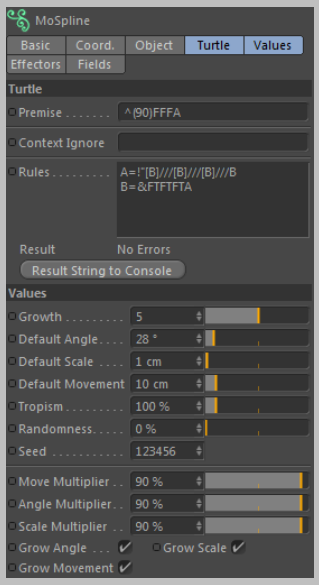
  - *Random Effector*
    - *Tab Effector* slúži na nastavenie sily efektoru a vybratie typu šumu
    - *Tab Parameter* určuje transformáciu ktorá sa v náhodné násobí 0..1 a aplikuje na klony
    - *Tab Deformer* určuje či sa deformácia prejaví len na klony (Off)alebo na objekty, polygóny či body(používa sa ako klasicky *Deformer*)
    - *Tab Falloff* určuje úpadok sily *Effektoru*
  - *Formula Effector*
    - *Tab Effector* slúži na nastavenie sily *Effektoru* a napísanie funkcie podľa ktorej sa vypočítavá sila transformácie(získaným číslom sa násobí transformácia pred aplikovaním na klon)
    - *Tab Parameter* ako u *Random Effektoru*
    - *Tab Deformer* ako u *Random Effektoru*
    - *Tab Falloff* ako u *Random Effektoru*
  - *Shader Effector*
    - *Tab Effector* slúži na nastavenie sily *Effektoru*
    - *Tab Parameter* ako u Random *Effektoru*
    - *Tab Deformer* ako u Random *Effektoru*
    - *Tab Falloff* ako u Random *Effektoru*
    - *Tab Shading* určuje *Shader* podľa ktorého sa určuje sila efektoru(podľa odtieňu čiernej 0..1)
    - Mapovanie vybratého shaderu je nie vždy dobre
    - Funguje pri globálnych efektoch ako šum ale nie pri mapovaní konkrétnych textúr, tie treba extra použiť a potom vybrať v nastavení *Channel* nie *Custom Shader* ale kanál namapovanej textúry ktorý chceme použiť a vybrať konkrétnu namapovanú textúru(jej tag)

MoInstance

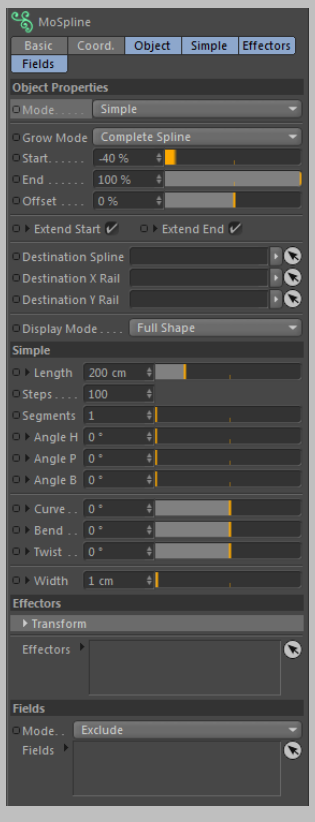


- Tento objekt sa nachádza v menu *MoGraph*
- Tento objekt vytvára klony podľa svojej vlastnej histórie vramci animácie
- Pozrie sa do minulosti kde sa nachádzal a vytvorí tam klon referencovaného objektu
- Ním vytvorene klony sú ako klasické klony *MoGraph*, čiže ovplivnitelné *Effektormi*
- Nastavenia :
    - Object tab obsahuje referenciu na objekt ktorý klonuje a akú hlbokú ma mať pamäť
    - *Transform* ta určuje transformáciu objektu pred klonovaním
    - *Effectors tab* je zoznam *Effektorov* ktoré pôsobia na tento objekt

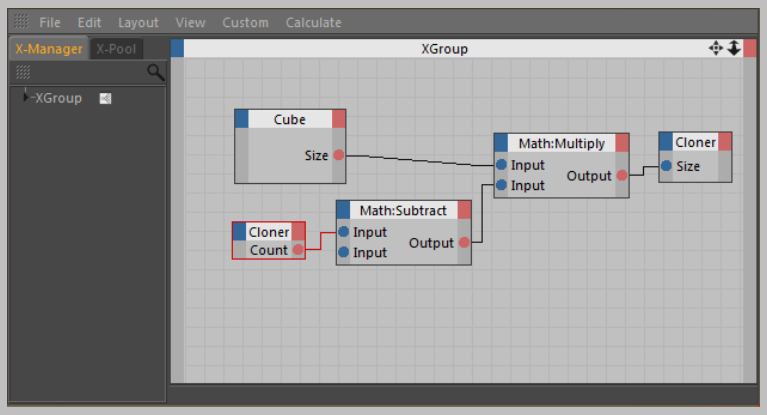
MoSpline



- Tento objekt sa nachádza v menu *MoGraph*
- Je to generátor kriviek
- Slúži na vytváranie Eulerových špirál z kriviek, rast kriviek, či ako generátor kriviek podľa parametrov
- Môže byt ovplivnovaný *Effektormi* ale aj poľami na pracú s jednoduchými časticami
- Simulácia rastu krivky je pomocou posuvníkov na začiatok (*Start*) a koniec(*End*) krivky
- Ma 3 módy prepínateľné v tabe *Object : Simple, Spline, Turtle*
    - *Simple mode*
        - Slúži ako generátor kriviek podľa parametrov
        - Nastaviteľný počet kópií a ich transformácia obohatená o rôzne zakrivenia
        - Všetko nastaviteľné v tabe *Object*
    - *Spline mode*
        - Slúži na úpravu krivky pre animovanie
    - *Turtle mode*
        - Tento mód slúži ako L-system
        - V *tabe Turtle* sa napíšu pravidla
        - V *tabe* Values sú hodnoty ovplivnujúce rast krivky ako veľkosť kroku defaultné hodnoty otočenia a veľkosti či náhodnosť kroku
        - *V Help* sa nachádzajú možné príkazy pre L-system



XPresso



- Je systém na vytváranie závislosti medzi objektmi a ich vlastnosťami
- Vytvára sa ako tag aplikovaný na objekt
- Slúži na animovanie či sprehľadnenie prace
- Je založení na uzloch *(Nodes*) a ich interfacoch medzi ktorými sa jednoduchým pretiahnutím z výstupného do vstupného vytvorí závislosť
- Novy uzol vytvorím kliknutím pravým tlačidlom na voľne miesto a vyberieme *New Node* a tam *XPresso*(umiestnenie základných uzlov)
- Kliknutím na ľavý(vstupný) či pravý(výstupný) horný štvorček nám ukáže interface a môžeme si vybrať ktoré časti z neho chceme použiť
- Je niekoľko kategórii uzlov
    - *General* sú základne uzly všeobecné zamerane(na pracú s objektmi či polygónmi, krivkami)
    - *Adapter* tu sú uzly na prevod medzi typmi hodnôt
    - *Bool* na booleovské operácie
    - *Calculate* na matematické operácie
    - *Script* na písanie vlastných uzlov cez Coffe alebo Python
    - *Logic* na logické operácie
    - *Iterator* ak potrebuje iterovať nejakú operáciu na číslach či v hierarchii objektov
- Často používaný uzol *Formula* používa označenie vstupných premenných formát „$x“ kde x je poradie vo vstupnom interface

Thinking Particles

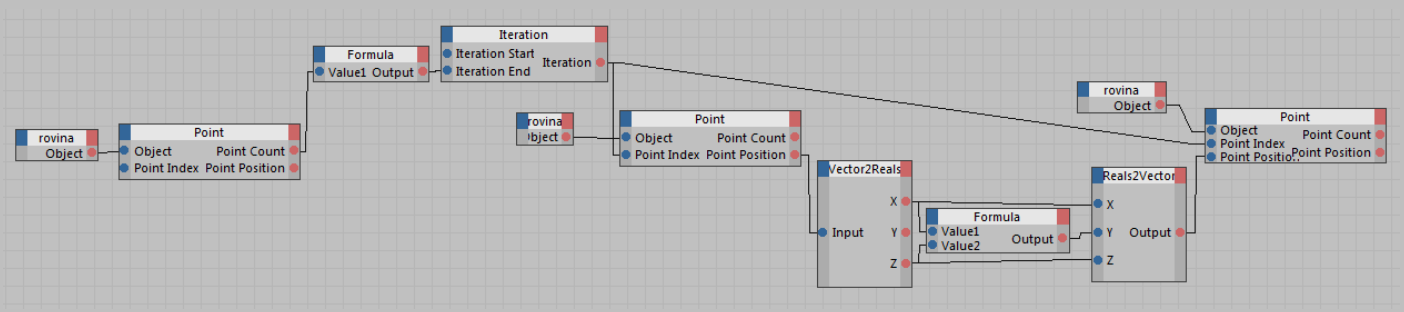
- Toto je systém na pokročilé simulácie častíc
- Častice sú reprezentovane bodmi
- Doporučujem využívať už predvytvorené objekty z knižnice *Studio-Simulation-Thinking Particles*
- V knižnici sú rôzne *Emitory, Effektory* a rôzne mechanizmy na interakciu
- Pre pokročilú simuláciu sa dajú programovať nove objekty pomocou *XPresso*
- Častice sú organizovane do *Particle Groups* (menu *Simulate-Thinking Particels-Thinking Particles Settings*) a podľa nich sa dajú nastavovať vplyvy efektov
- Nato aby sa častice zobrazili, ak vytvárajú nejaké objekty, nielen body, treba *Particle Geometry* (menu *Simulate-Thinking Particles- Particle Geometry*) ta slúži na zobrazenie konkrétnej skupiny častíc

Generátory

- V tejto kapitole sa nachádza predstavenie rôznych generátorov ktoré sa dajú vytvoriť v Cinema 4D
- Generátory kriviek, objektov, podľa presných pravidiel či náhodné
- Prechádza sa od kriviek, cez objekty až po svetlo

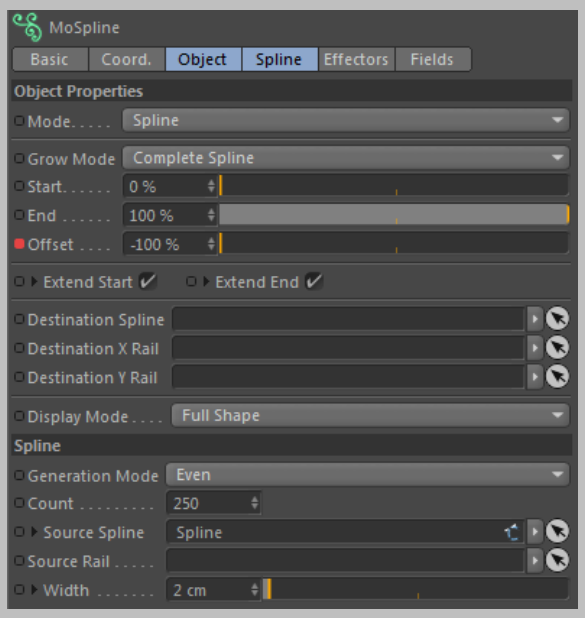
Matematické objekty

- Na krivky je najlepšie použiť formula krivku
  - Vytvoríme *Formula* (nachádza sa medzi štandardnými krivkami)
  - V nastavení môžeme písať rovnice pre x,y,z súradnicu *X(t),Y(t),Z(t)* každého bodu podľa parametru (zápis ako klasická parametrická krivka)
    - Napríklad *X(t)=t*t , Y(t)=sin(t)*50 , Z(t)=cos(t)*100*
  - Nastavenia *Tmin* a *Tmax* nám určujú interval v akom sa pohybujeme s parametrom
    - Napríklad Tmin=0 , Tmax=1
  - Hodnota *Sample* určuje koľko bodov krivky vzniká, koľko krokov parametru je vramci intervalu
    - Napríklad *Sample=30*
- Na rovinu treba komplikovanejšie, dá sa iba explicitná rovina(y je závislé na x a z)
  - Vytvoríme *Plane* (nachádza sa medzi štandardnými objektmi) a nastavíme koľko chceme polygónov aby mala (dôležité pretože následné ovplyvňujeme body týchto polygónov)
    - Napríklad Width Segments=40 , Height Segments=40 , veľkosť môžeme nechať pôvodnú
  - Vytvoríme *XPresso tag* na tejto rovine
  - Do neho si prenesieme rovinu (kliknúť, držať a preniesť do *XPresso Editora*)
  - Vytvoríme uzly *Point(Xpresso-General), Iteration(XPresso-Iterator), Formula(XPresso-Calculate)* a adaptéry *Vector2Real a Reals2Vector (XPresso Adapter)*
  - Vytvoríme kópie a vytvoríme a spojíme interface podľa obrázku



  - Láva formula je nastavená na *Formula=$1-1*
  - Pravú formulu nastavíme podľa funkcie ktorá generuje rovinu
    - Napríklad *Formula=sin($1*$2/pi)*10*

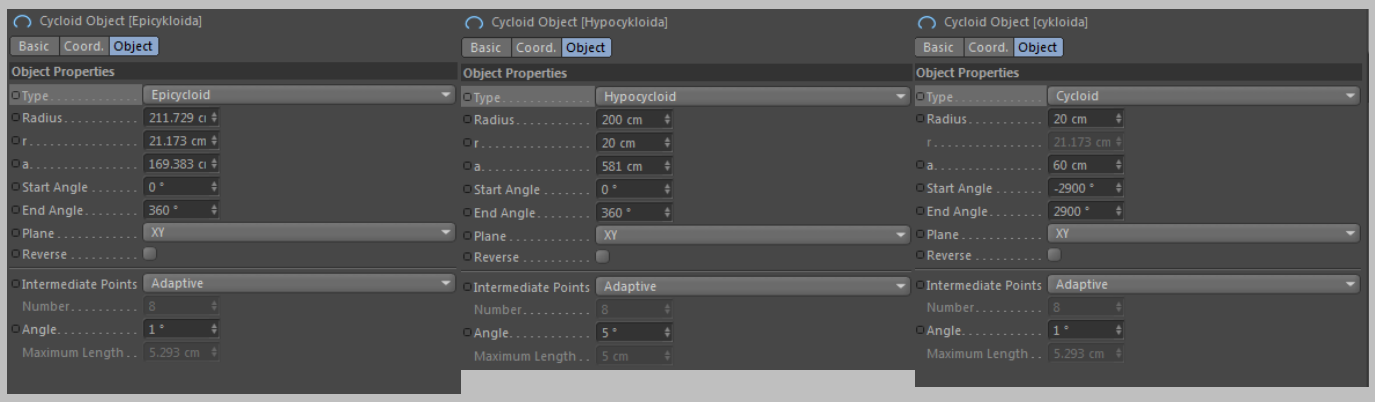
Eulerove špirály



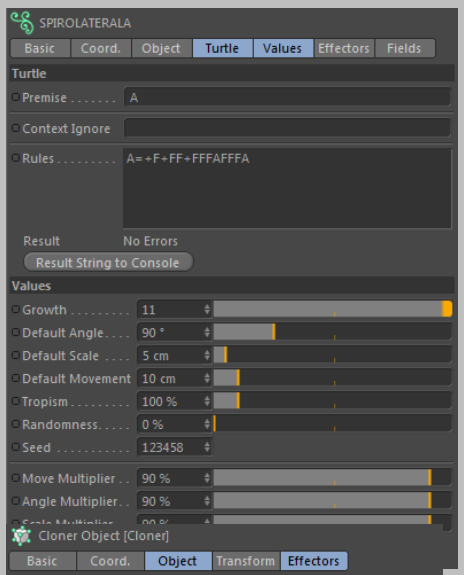
- Vytváraná buď ako parametrická krivka alebo pomocou *MoSpline* mód *Spline*
- Ukážeme si pomocou *MoSpline*
    - Vytvoríme si *MoSpline* a prepneme mód na *Spline*
    - Vytvoríme nejakú zdrojovú krivku
    - Zdroj vložíme do *Source Spline (Mospline-tab Spline)*
    - Generation mode zmeníme na *Even* (pre lepšiu segmentáciu)
    - *Count* zmeníme na požadovaný počet segmentov krivky
        - Napríklad *250*
    - Ako pohybujeme hodnotou *Offset* tak nám vznikajú Eulerovské špirály na koncoch krivky
    - Rovnako efekt dosiahneme ak nastavíme *Start* či *End* mimo rozsah *0..1*

Cykloidy

- Vytváraná pomocou štandardnej krivky *Cycloid*
- Môžeme vytvárať rôzne krivky výberom Typu v nastavení cykloidy*(Cykloidy, Epicykloidy, Hypocykloidy)*
- Ukážky rôznych nastavení parametrov:

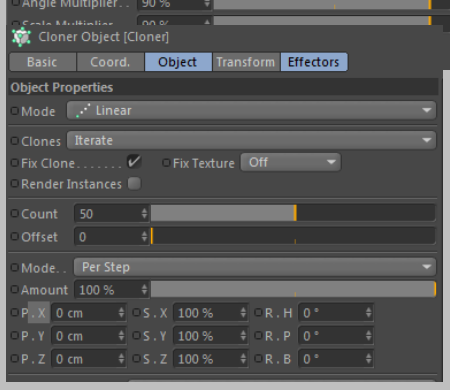


Spirolateraly



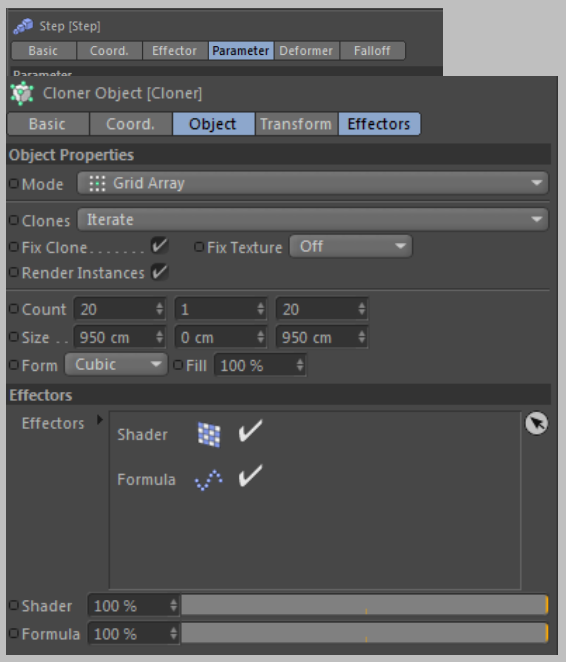
- Vytvárané pomocou L-systemu v *MoSpline* mód *Turtle*
- Vytvoríme *MoSpline* a nastavíme mód *Turtle*
- Na vytvorenie Spirolateraly nastavíme *MoSpline* nasledovne
- V tabe *Values* nastavíme *Default Angle=90, Growth=10* (počet iterácii, pre lepšie zobrazenie)
- V tabe *Turtle* nastavíme *Premise=A* *a Rules=A=+F+FF+FFFAFFFA*
- Týmto získame spirolaterálu v 10 iterácii

Opakovane Transformácie



- Vytvárané pomocou *Clonera* a *Effektoru Step*
- Vytvoríme *Cloner* v *Linear mode*
- Vynulujeme transformáciu v kroku
- Vytvoríme *Step Effector* a vložíme ho do *Effectors Clonera*
- Nastavíme konečnu transformáciu ktorá sa interpoluje podľa krivky v tabe *Effector*
- Pozíciu ďalšieho klonu získame aplikovaním transformácie na predchádzajúci
- Ukážkové parametre na obrázku

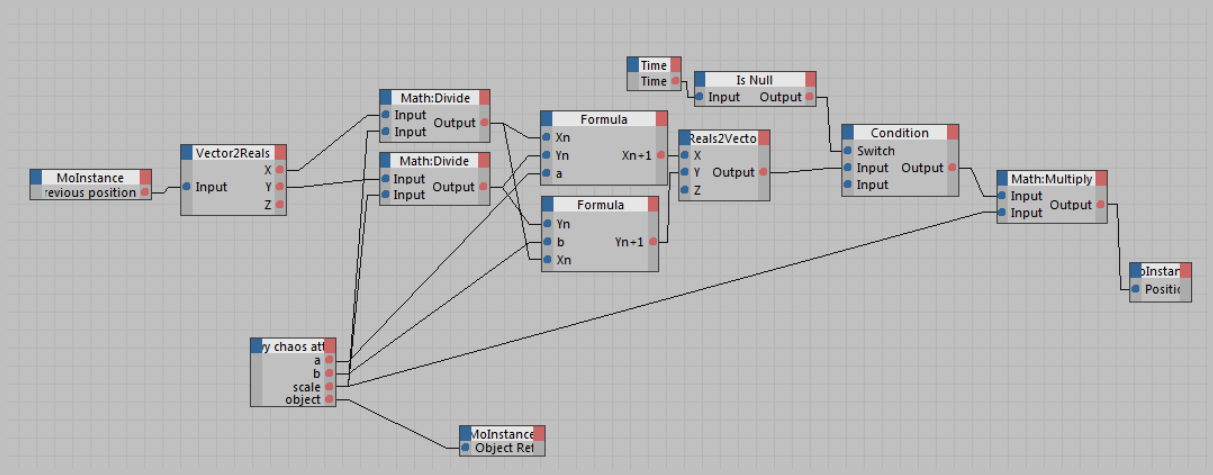
Transformácie na matici



- Vytvárané pomocou *Clonera* a rôznych *Effektorov*(podľa vlastnej chuti
- Vytvoríme *Cloner* v *Grid mode*
- Nastavíme počet v ose y na 1, aby sme dostali maticu (použiteľné aj s 3d maticou)
- Necháme klonovať ľubovoľný objekt
- Pridaním *Shader Effektoru* môžeme ovplyvňovať výšku podľa obrázka
- *Formula Effector* môže vytvárať periodické zmeny v klonoch

Chaotické atraktory

- Princíp ich tvorby leží v objekte *MoInstance*
- Pomocou *XPresso* si vypočítavame v čase zmenu z ich predchádzajúceho stavu
- *MoInstance* nám potom vie na konci zobraziť históriu celého výpočtu, čiže výsledný attraktor
- Nebudem tu rozoberať detailne jeho tvorbu , môžete si ju prezrieť v priložených príkladoch
- Vo *Viewport* sú vytvorene odkazy na hodnoty parametrov
- Inak parametre sa nastavujú grupujúcom *Null* objekte v tabe *User Data*



Dynamické systémy častíc

- Simulácie častíc
- Obeh planét okolo slnka , polystyrén a ventilátor, všetko plne simulovateľné
- Základ spočíva vo vyberaní správnych *Emitorov* a *Effektorov* vramci *Thinking Particles* knižnice
- Alebo vo využití základného časticového systému(nachádza sa vramci *Simulate-Particles* menu)
- Už je len na vás ako si jednotlivé komponenty poskladáte
- *Thinking particles* sa dajú aj programovať pomocou *XPresso* ale toto sú už pokročilé techniky neobsiahnuté v tomto prehľade

Caustics

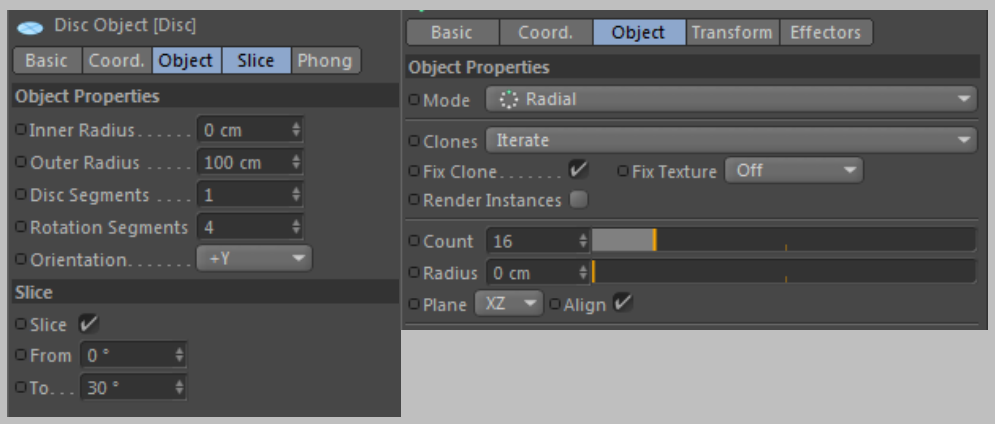
- Simulácia lámania svetla na povrchu či objeme priehľadných objektov
- Je potrebne vytvoriť svetlo zapnúť mu príslušný typ kaustiky na povrchu alebo objeme
- Taktiež je potrebne pridať tento efekt a zapnúť v nastavení renderu
- Pri vzniku krúžkov svetla namiesto plôch treba zvýšiť počet fotónov v nastavení kaustiky vo svetle

Mozaiky

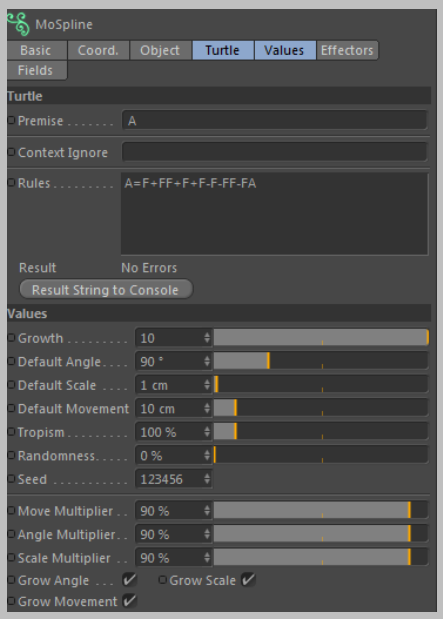
- V tejto kapitole si predstavíme mozaiky, ako ich generovať či uľahčiť ručné vkladanie
- Budeme hlavne používať *MoGraph, XPresso* a *Snapping*
- V príklade Základne operácie nájdete vytvorený systém na základne grupy symetrií pre mozaiky so štvorcovým razidlom(ovládanie cez *User Data* v *Null* objekte)

Rozetové mozaiky

- Základ razidla je dobre si vytvoriť z objektu *Disk*
    - Segmentáciu si nastavíme podľa potreby
    - V tabe *Slice* si nastavíme uhol výrezu aký chceme (Cinema vie spracovať aj matematické operácie čiže v tabe *Slice* môžeme napísať aj *From=0 , To=360/12)*
- Takto získané razidlo môžeme ďalej upravovať a pridávať ďalšie modely, ktoré vložíme do hierarchie pod neho
- Ak chceme dihedrálne mozaiky , vložíme toto razidlo do *Symetry* a počet dielov pri kopírovaní sa zmenší na polovicu
- Vytvoríme *Cloner* a toto razidlo vložíme do
    - Nastavíme mód na *Radial*
    - Nastavíme *Radius* na *0* a *Count* na počet klonov koľko treba aby vznikol z výrezov kruh
- Týmto sme získali cyklickú či dihedrálnu mozaiku podľa toho či sme aplikovali symetriu alebo nie



Pásové mozaiky

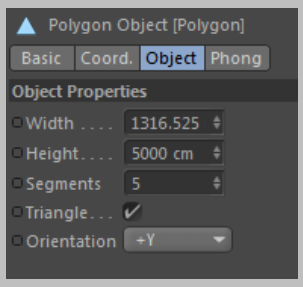


- Môžeme si ich predstaviť ako kopírovanie razidla vedľa seba a aplikovať transformácie naň
    - Na toto stačí *Cloner* a pomocou *Effektorov Formula* s vzorcom *mod(id;2)* môžeme otáčať a robiť symetrie
    - Príklad Základne operácie
- Alebo ako L-system kde generujeme rovinnú krivku s požadovaným vzorov a potom ju ďalej upravíme
    - Vytvoríme si *MoSpline* v režimu *Turtle*
    - Napíšeme si pravidla tak aby v jednej iterácii som sa z pozície (0,0,0) dostal na pozíciu (n,0,0) s rovnakou rotáciou, kde n je dĺžka jednej iterácie
    - Týmto môžeme iterovať koľko chceme a dostaneme pásovú mozaiku
    - Napríklad: *Premise=A , Rules: A=F+FF+F+F-F-FF-FA*

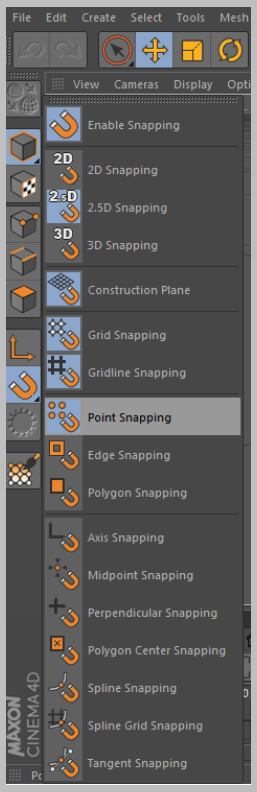
Tapetové mozaiky

- Základný nastroj na ich tvorbu je *Cloner* v spolupráci s *Effektormi* a prípadným *XPresso* pre zjednodušenie
- Poriadne si poprezerajte priložené príklady(Základne dláždenia), niektoré sú už použiteľné na ďalšiu tvorbu
- Periodické mozaiky
    - Vytvorte *Cloner* v režime *Grid Array*
    - *P*ri štvorcových stačí len klonovať
    - Pri trojuholníkových treba pomocou *Effektoru Formula* (vzorec *mod(id;2) R.H=180)* otočiť každý druhy trojuholník o 180
    - Pri šesťuholníkoch
        - Treba klonovať podľa nie celkovej dĺžky ale tak aby po posunutí každého druhého dole do seba zapadali
        - Treba pomocou *Effektoru Formula* (vzorec *mod(id;2) P.Z=1/2*z* | kde z je veľkosť razidla v ose z) posunúť šesťuholníky aby do seba zapadali
        - Sú dve možnosti ako klonovať ,
        - Buď bude počet klonov v ose X bude vždy parný aby sme mohli každý druhy posúvať
        - Alebo musíme veľkosť mriežky v ose Z zmenšiť na polovicu a potom nám nevadí aj neparný počet klonov v ose X ale dochádza k zdvojeniu niektorých klonov

- Neperiodické mozaiky
  - Spomenieme si Špirálové, polymorfne, hierarchické
  - Špirálové
    - Vytvoriť objekt *Polygon* so zapnutým *Triangle* týmto vzniká trojuholník ktorý môžeme pomocou *Clonera* klonovať radiálne
    - jeho segmentáciou cez nastavenie *Segments* vznikajú rôzne úrovne dláždenia
    - Je potrebne doladiť veľkosť polygónu tak aby jeho klonovaním vznikol kruh (napr. obrazok)

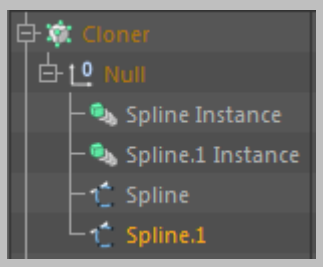


  - Polymorfne
    - Je potrebne si vytvoriť nejaké razidlo ktoré sa dá klonovať pomocou *Clonera*
    - Vytvorenie a následné dláždenie pomocou Clonera tu už bolo prebraté a tak sa mu nebudeme venovať znova
    - Je možné si bližšie prezrieť príklad Neperiodické polymorfne dláždenie
  - Hierarchické
    - Buď si ručné vytvoríme rôzne úrovne dláždenia alebo môžeme pomocou *Clonera* a *XPresso* si zostrojiť automatické nastavenie úrovne podľa posuvníka čo nám zlachší dláždenie
    - Pre konkrétne ukážky a nastavenia si môžete pozrieť príklad Neperiodické Hierarchické mozaiky



- Aperiodické mozaiky
  - Tu sa nedá inak ako ručné dláždiť
  - Ale ak mame presne vytvorene dlaždice tak môžeme použiť funkciu *Snapping*
  - Nachádza sa na vľavo od *Viewportu,(magnet)*
  - Keď klikneme tak zapneme a vypneme funkciu
  - Podržaním na ikone môžeme meniť režim pripínania
  - Na dláždenie je vhodný *Point Snapping* čím môžeme roh jednej dlaždice presne premiestniť na roh druhej

Esherove mozaiky



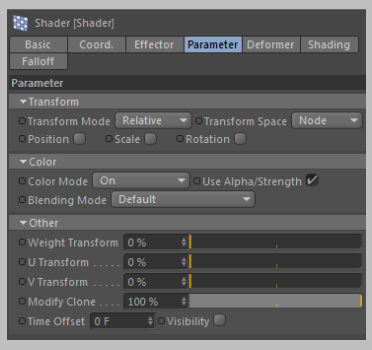
- Dláždenie je ľubovoľné ale tvorba razidla je špecifická
    - S tým si môžeme pomôcť tým že si z základného tvaru napríklad štvorca
    - Vytvoríme 2 krivky z 2 strán štvorca
    - Potom vytvoríme *Instance* týchto dvoch kriviek
    - Instancie umiestnime tak aby sme krivky s instanciami mohli použiť ako razidlo pri klonovaní
    - Viz. príklad
- Esher sa tiež zaoberal stuhovými mozaikami
    - Pri vytváraní razidla pre ne je dobre ho robiť podľa *Plane* (segmentovaný 4x4)ktoremu priradíme tag *Display* a vňom zapneme prvý príznak a vyberieme *Shading mode = Lines*
    - Týmto získame mriežku so zvýraznenými vstupmi a výstupmi stuh
    - Podľa takejto mriežky už ľahko vytvoríme jednotlivé stuhy ako krivky a keď zapneme *Snapping mod Point Snapping* tak presne a rýchlo môžeme kresliť stuhy vramci razidla

Islamské mozaiky

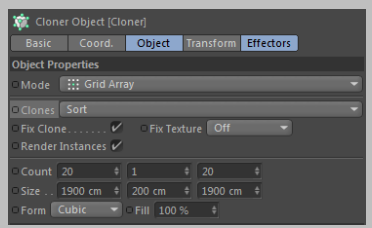
- Môžeme si vytvoriť pomocnú konštrukciu a podľa nej designovať mozaiky(viz príklad)
- Alebo si ich vytvoríme v iných programoch a len prenesieme do Cinemy
- Ako obrázok ktorý si môžeme vložiť do pozadia *Viewportu* v pohľade *Top* a ručné vytvoriť krivky pre každú časť, alebo importovať priamo krivky a pracovať s nimi

Ďalšie spôsoby konštrukcie

- Tieto metódy sa v Cineme nedajú tak dobre automatizovať
- Tak treba mozaiky vymyslieť a potom ich vytvoriť v Cineme podľa obrázka či krivky
- Na 5-uholniky sa dajú použiť konštrukčné nástroje ktoré presne zisťujú uhol a vzdialenosť(viz príklad Provokujúce 5-uholniky)



- Alebo naisť a využiť nejakú pravidelnosť v nich (Viz príklady Členené dláždenie)
- Na modulárne mozaiky sa dá pozerať ako na obyčajné mozaiky ale s viacerými razidlami
    - Razidla môžeme vyberať náhodné – v *Cloner* parameter *Clones=Random*
    - Podľa farby klonu – v klonery *Clones=Sort*
        - Na *Cloner* musí byt aplikovaný aspoň jeden *Effector*
        - V tomto *Effectore* musí byt nastavený v tabe *Parameter* podmenu *Other* parameter *Modify clone=100%*
        - Takto sa vyberie razidlo podľa odtieňu šedej ktorý dal tento *Effector* klonu (viz príklad Modulárne mozaiky)



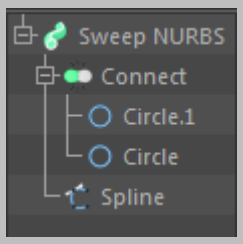
Neeuklidovské mozaiky

- Je potrebne si nasimulovať neeuklidovskú projekciu
- Nie je to dokonala projekcia ale ako simulácia je to dostačujúce
- Hyperbolická projekcia
    - Vytvoríme si kameru a umiestime ju do bodu (0,0,0) s rotaciou(0,90,0)
    - Vytvoríme projekčný disk buď odhadom ako *Lathe Nurbs* podľa vlastnej krivky alebo si vytvoríme explicitne definovaný povrch
    - Nastavíme *Render* aby bol štvorcový, nemusíme ale najvhodnejšie na projekciu
    - Obidva prístupy , explicitne definovaný povrch či *Lathe Nurbs,* majú svoje mapovania textúr
    - Mapovanie textúr veľmi ovplivnuje výsledok, musíme správne zvoliť, viz. príklad

Uzly

- V tejto kapitole si ukážeme ako vytvárať uzly a ako ich ďalej upravovať
- Designovanie uzlov je obmedzene v Cineme na matematické, ručné alebo podľa obrázka
- Úprava prebieha pomocou *Clonerov* a *Nurbs* povrchov

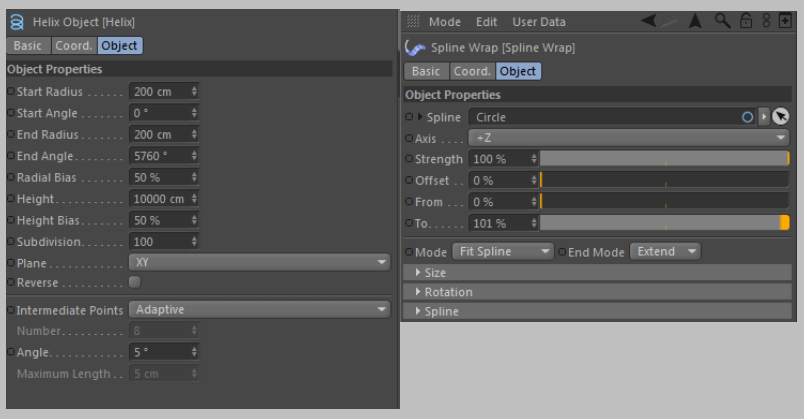
Rotačné lano



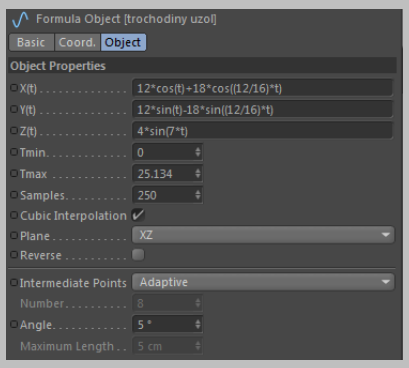
- Tento typ uzla je jednoducho vytvoriteľný
- Je to vlastne *Sweep Nurbs*, kde krivku po ktorej ide je akákoľvek
- A prierez si môžeme spraviť tiež akýkoľvek, čiže aj cez objekt *Connect* spojiť viacero profilov dokopy
- V nastavení *Sweep Nurbs* je možnosť nastavenia rotácie cez *End Rotation*
- Týmto sa nám profil bude pekne rotovať po krivke a vznikne rotačné lano
- Príklad je spojený s prstencovým uzlom

Prstencový uzol

- Tento typ uzla vzniká po aplikovaní *Deformeru Spline Wrap* na *Helix* kde *Spline Wrap* ma ako zdroj kružnicu
- *Spline Wrap* musí mat osu *Axis* doplnkovú k osiam helixu
- Zaujímavé je aj sa pohrať s parametrami *Spline Wrap,* obalenie nemusí byt unimorfne a môže doisť k ďalším transformáciám
- Je potrebne aby *End Angle* v *Helix* bol násobok 360 aby sa po obalení po kruhu *Helix* vrátil spať , ako keby uzavrel krivku
- Po tomto vytvorení krivky môžeme aplikovať princíp rotačného lana a vznikajú zaujímavé útvary ako v príklade Rotačný Prstencový uzol



Trochoidny uzol



- Trochoidné uzly sú vhodne na spracovanie v Cineme, pretože sú matematicky dobre definovateľné
- Stačí si zistiť rovnice uzlu a použiť *Formula* krivku na vytvorenie tohto uzla
- Je potrebne niekedy násobiť konštantou vzorec aby bolo generovaná prijateľné veľká krivka
- Vzorce pre každú osu pracujú s radiánmi

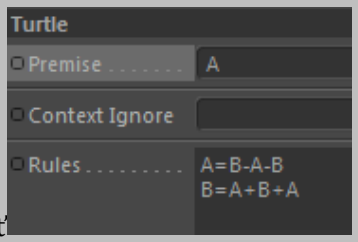
Dokončovanie uzlov

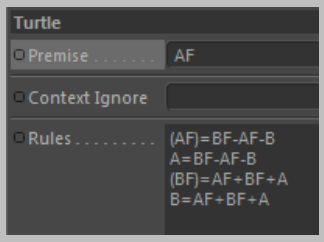
- Uzly vytvorene v iných programoch
- Ak sú importovane ako krivky tak je to bez problému
- Ak ako obrázok tak je potrebne uzol ručné vytvoriť, najskôr ako krivku
- Potom pomocou *Sweep Nurbs* a rôznych profilov dať tomuto uzlu tvar ako polygonálny objekt
- Taktiež môžeme po tejto krivke generovať objekty pomocou *Clonera* v mode *Object*
- Priložil som ako príklad moju finálnu pracú Highway to nowhere
    - Uzol som vytvoril v programe KnotsBag a exportoval ako obrázok
    - Obrázok som si vložil do pozadia v pohľade *Top*
    - Ako som postupne vytváral krivky uzla tak som si predchádzajúce body pre každú časť krivky zdvihol alebo znížil tak aby pri ďalšom krížení som nemal dva body na sebe v jednom mieste
    - Po ručných úpravách aby všetko sedelo som mal hotové krivky
    - Vytvoril som si profil cesty a ten som dal do *Sweep Nurbs* s krivkou uzlu
    - Pomocou *Clonera* som generoval lampy , poklopy na ceste, smetiaky a podporne stĺpy
    - Už zostávalo len otexturovať objekt a dotvoriť scénu a je hotovo
    - Objekty lámp a poklopov a materiály sú prevzaté z knižnice *Visualize*
- Taktiež pre použitie do budúcna môžeme uzly upraviť pomocou *XPresso*
    - Napríklad u ostnatého drôtu aby sa helix dostatočné skrúcal, a generovaný bol správny počet ostňov

Fraktály

- Generovanie 2D fraktálov je ponechané na špecializované programy
- V Cineme ale môžeme vynikajúco vytvárať Fraktály pomocou L-systemov a ako iterované funkcie na objektoch
- Taktiež Fraktály sú používané pri generovaní šumu

L-system Fraktály

- Základ je *MoSpline*
- Záleží už len na pravidlách
- L-system v Cineme pracuje s vlastnými pravidlami
- Nemôžeme si definovať viac písmeniek ktoré robia krok dopredu
- Môžeme ale použiť komplikovanejšie pravidla, nie sú práve elegantne ale dovoľujú nám definovať viac písmeniek ktoré robia krok dopredu napríklad
  - Musíme zmeniť 

na 

  - *A -> FA,* na ľavej strane *A->(FA)* , ak je to posledné písmenko tak *A->A* a pridáme duplikát ktorý na ľavej strane nemá F
  - Prvý obrázok negeneruje nič pretože sa neobjavuje ani jedno písmenko na pohyb, v druhom transformovanom už áno
- Sú priložené príklady
  - Gosperova krivka
  - Hilbertova krivka
  - Kochova vločka
  - Sierpinskeho trojuholník

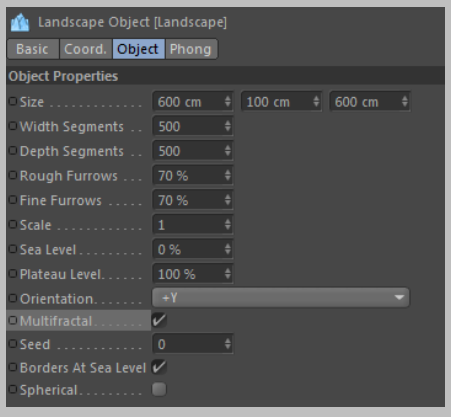
F(a,b,c)	Moves the Turtle forwards and optionally draws lines along its length (a), for scale (b) and/or subdivision (c).
f(a)	Moves the Turtle forwards and NO line will be drawn (i.e. a Spline segment will be ended). „a" is the optional length value.
H(a,b,c)	Moves the Turtle forwards and optionally draws a line half as long as its length (a), its scale (b) and/or its subdivision (c)
h(a)	Moves the Turtle forwards and NO line will be drawn (i.e. a Spline segment will be ended). „a" is the optional half length value.
M(a,b,c)	Moves and draws a line. The Turtle will move within local coordinates but will itself not rotate.
m(a,b,c)	Turtle moves forward but does not draw a line. The Turtle moves within local coordi nates but will itself not rotate.
G(a)	Turtle moves forward but does not draw a line and does not end a Spline segment. „a" is the optional length value.
R	Resets the Turtle back to the segment starting point and draws a line (in a MoSpline Turtle a segment is always contained in [ ]).
R	Resets the Turtle back to the segment starting point and does not draw a line (in a MoSpline Turtle a segment is always contained in [ ]).
P(x,y,z)	Places the Turtle at these coordinates and draws a line.
p(x,y,z)	Places the Turtle at these coordinates and does not draw a line.
-(a)	Rotates the Turtle counterclockwise around the vertical axis. „a" is the optional angle value.
+(a)	Rotates the Turtle clockwise around the vertical axis. „a" is the optional angle value.
&(a)	Rotates the Turtle forwards around the transversal axis. „a" is the optional angle value.
^(a)	Rotates the Turtle backwards around the transversal axis. „a" is the optional angle value.
/(a)	Rotates the Turtle clockwise around the center axis. „a" is the optional angle value.
\(a)	Rotates the Turtle counterclockwise around the center axis. „a" is the optional angle value.
[	New branch = new Turtle
]	Branch ends
{	Polygon begin (see example below: „Polygon creation")
}	Polygon end
.	Adds a polygon point (within curly brackets for polygons: e.g. = {.-f.+f.+f.+f}).
|	Rotates the Turtle 180° around the vertical axis.
*****	Rotates the Turtle 180° around the center axis.
%	Prunes the branch at this point (everything will be ignored through to the end of the close bracket:]; see example „Prune branch").
"(a)	Multiplies the length by the optional „a" value for each generation (see example below „Example: Multiply/divide).
!(a)	Multiplies the scale by the optional "a" value for each generation (MoSpline diameter when used in conjunction with SweepNURBS).
;(a)	Multiplies the angle by the optional value „a" for each generation.
_(a)	Divides the length by the optional value „a" for each generation.
?(a)	Divides the scale by the optional value „a" for each generation (MoSpline diameter when used in conjunction with SweepNURBS).
@(a)	Divides the angle by the optional value „a" for each generation.
T(a)	Adds Tropism (e.g. FTFTF), which means that each Spline segment will curve slightly in the direction of the global Y axis. Any particle modifiers present will be avaluated.
$(x,y,z)	Displays the end of an Up Vector and orients the Turtle accordinly. Define a vector in whose direction the Turtle should face.
I(a,r,g,b) **J(a,r,g,b)** **K(a,r,g,b)** **L(a,r,g,b)**	(First character string begins with a capitol „i"). Here, any number of clones generated by a Cloner object can be arranged along the Turtle Spline. „r,g,b" represents the color of the clone. „a" is the index (i.e. the number that represents the order in which an object is arranged below a Cloner - see „Index" below). I, J, K, L represent the respective *Groups* in the **Cloner** object's **Mode** parameter, „Group 1", „Group 2"„Group 3"„Group 4". An example is given below under „Example: Cloner object".

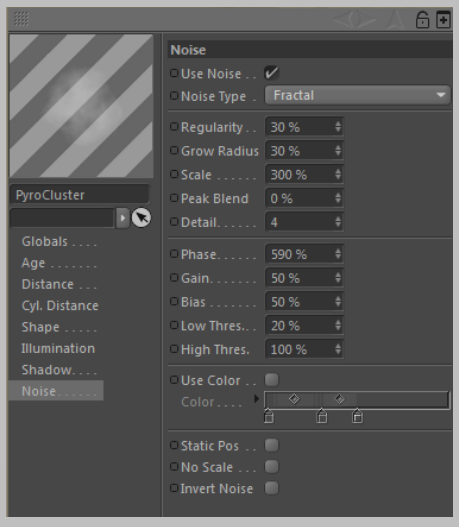
Iterované Fraktály

- Základ je *Cloner*
- Vysvetlene na Mengerovej špongii
    - V Mengerov špongii sa iterovaním kocka mení na kocku ktorá je ako keby rozložená na 3*3*3 menších kociek niektoré z nich sú vymazane (Mengerova špongia ich ma definovane ale iterovať sa dá s akýmikoľvek)
    - Vytvoríme si pomocnú kocku ktorú použijeme na klonovanie
    - Pomocná kocka ma 2/3 veľkosť pôvodnej
    - Menšie kocky majú 1/3 veľkosť pôvodnej
    - Vytvoríme na pomocnej kocke nove body v stredoch hrán(nástrojom *Knife* to ide výborné)
    - Vytvoríme si *Cloner* v mode *Object* a odškrtneme *Align clone* nastavenie v tabe *Object*
    - Vložíme menšiu kocku do *Clonera* a nastavíme mu *Source Objekt* na pomocnú kocku
    - Týmto sme získali pôvodnú kocku zloženú z menších
    - Teraz musíme rekurzívne spraviť ďalšiu iteráciu tým proces zopakujeme na menšej kocke a výsledok zostane v hierarchii objektov na mieste pôvodnej
    - Viz príklad Menger Sponge

Fraktálové šumy

- Používajú sa v generovaní terénov a pri generovaní šumu v module *PyroCluster*
- Pri *Terain* objekte sa zapínajú príznakom *Multifractal* v tabe *Object*
    - Tento mód je vhodný na generovanie prírodných terénov
- Pri *PyroClustery* sa vyberá v nastavení materiálu podmenu *Noise* a tam *Noise Type*
    - Tento šum je vhodný na paru, oblaky a prírodné úkazy
- Viz príklad Fractal Noise

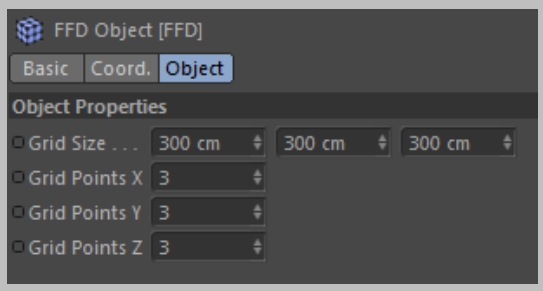




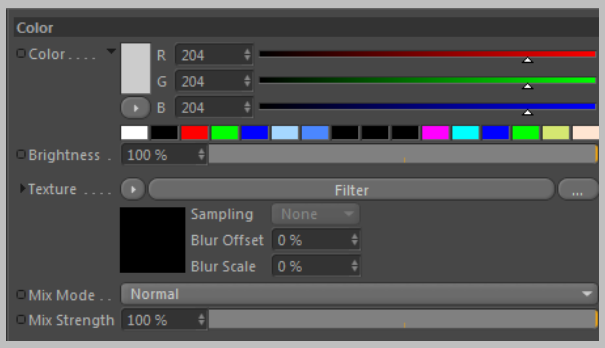
- V tejto kapitole si ukážeme transformácie, fylotaxiu a zlatú špirálu
- Na organic art sa dá použiť klasické modelovanie vylepšené o metabal či klonovanie

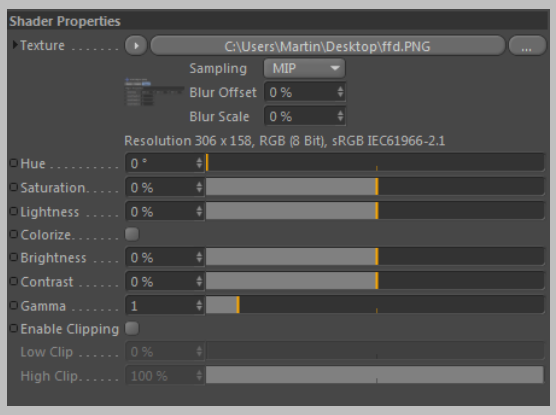
Transformácie

- Transformácie sa vytvárajú pomocou deformátorov
- Môžeme použiť klasické ako skosenie, otočenie, ohnutie
- Alebo vytvárať voľne deformácie pomocou *FFD* čo je Free-form Deformer
    - Týmto získame kockovú mriežku a úpravou bodov tejto mriežky deformujeme body ktoré sa nachádzajú vnútri mriežky
    - Pracuje sa sním ako s klasickým deformerom čiže musí byt v hierarchii pod deformovaným objektom alebo na rovnakej úrovni ale obidva musia byt pod nejakým objektom



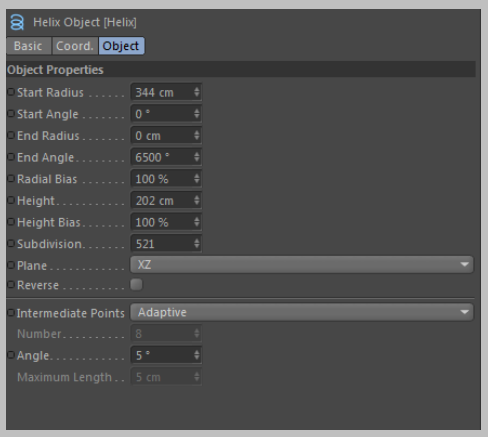
- Transformovať môžeme aj textúry
    - Či už zmiešavaným s druhými, úpravou UV mapy alebo pomocou *Efektu Filter*
    - Pri výbere textúry (*Texture*) v materiáli vyberieme textúru(obrázok či Shader) ktorý chceme upraviť
    - Potom vyberieme znova textúru ale tentoraz Shader z podmenu *Effect* a to *Filter*
    - Týmto sa nám naša pôvodná textúra aplikuje v Shaderi *Filter*
    - V tomto Shaderi môžeme meniť odtieň , farebnosť, svetlosť, kontrast, či orezávať bielu alebo čiernu





Zlata Špirála

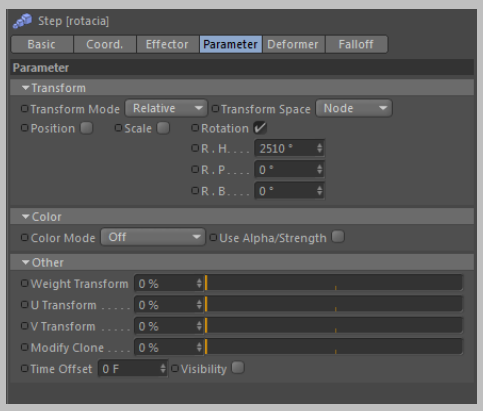
- Tuto špirálu môžeme vytvoriť buď ako parametrickú krivku, ak vieme vzorce, pomocou krivky *Formula*
- Alebo použijeme krivku *Helix*
    - Tam nastavíme *End Radius* = 0 , *Radial Bias* = 100%
    - Týmto sme získali základ pre zlatu špirálu
    - *Angle* upravuje ako veľa sa tato špirála zatáča
    - *Height* upravuje výšku tejto špirály
    - Height Bias upravuje rýchlosť stúpania



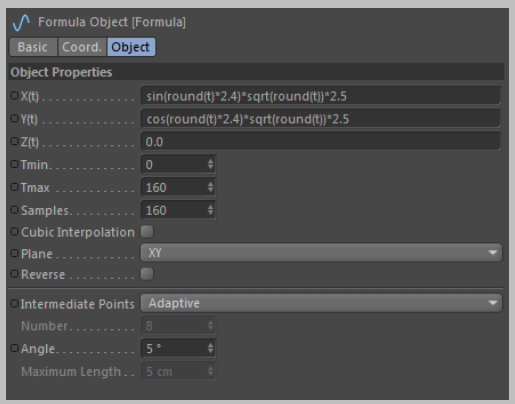
- Tuto špirálu následne môžeme vložiť do Sweep Nurbs s nejakým kruhovým profilom a získame ulitu
- Tento objekt môžeme ďalej upravovať pomocou textúr a deformerov
- Viz príklad

Fylotaxia

- Fylotaxia je rozmiestnenie listov či semiačok na rastline
- Ak generujeme listy na konári pomocou *Clonera*
    - Tak môžeme naňho použiť Effector *Step Effector* s príslušnou rotáciou
    - Ale pri zadávaní uhla rotácie musíme dať pozor pretože uhol zadaný sa rovná uhlu celkovému otočeniu, kde na nultý klon nie je žiadne otočenie aplikované



    - Čiže otočenie sa musí rovnať (n-1)*a | n je počet klonov, a je zmena otočenia
    - Viz príklad, kde je vytvorená závislosť cez XPresso vzhľadom na zadaný uhol a počet klonov
- Ak chceme generovať semiačka
    - Tie v prírode využívajú Fermatovej špirály
    - Táto špirála je matematicky dobre definovateľná čiže si ju môžeme vytvoriť ako krivku ktorá bude mat body tam kde chceme klon semiačka
    - Vzorce musíme upraviť tak aby namiesto spojitého parametru , ho zaokrúhľovali (*round(t)*)
    - Vzorkovanie krivky musí byt rovnaké aká je veľkosť intervalu parametru
    - Potom vytvoríme *Cloner* v mode *Object* ako zdroj udáme tuto krivku



    - Zmeníme *Distribution* na *Vertex toto* vytvorí klon v každom popisnom bode krivky(nie v tých ktoré vznikajú interpoláciou)
    - Viz príklad, objekt *FERMAT SPIRAL* a *Cloner* sú oddelene kvôli konfliktu so *Spline Wrap*

Op-Art

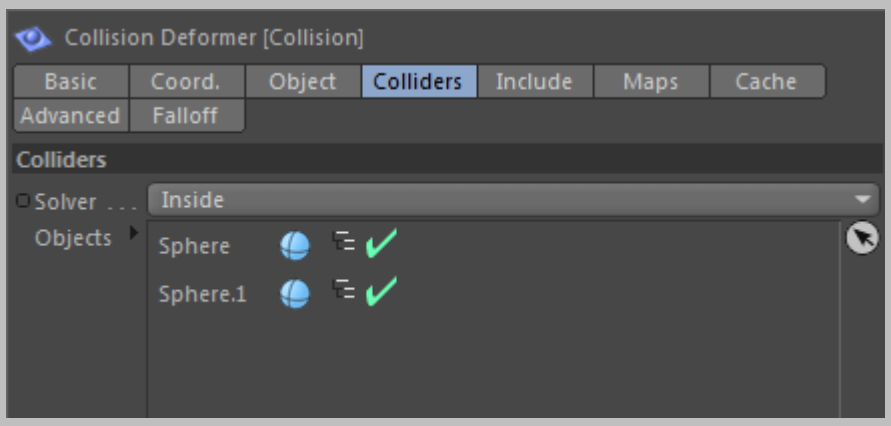
- Aj v Cineme sa dajú vytvárať zaujímavé optické ilúzie
- Dosť veľa je potrebné vytvoriť niekade inde a potom už len spracovať v Cineme
- Ale niektoré možno aj priamo v nej

Moiré vzor

- Moiré vzor vzniká z dvoch podobných mriežok ktoré sú mierne vychýlené od seba
- Mriežky môžeme vytvárať buď z textúry s alfa kanálom v ktorom je Shader *Tiles* či pomocou shaderu *Gradient*
- Alebo ako polygonálny model , buď klonovaním jednej časti mriežky alebo vymazaním niektorých polygónov z plôch ktoré sú zložené polygónov v mriežke ako *Plane* či *Disk*
- Viz príklad Moiré

Deformácie plochy

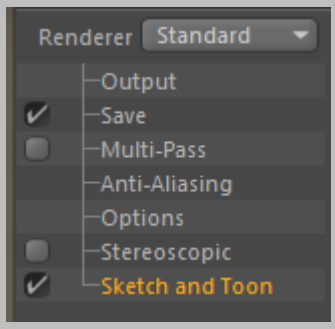
- Vytvoríme si plochu *Plane* na ňu aplikujeme šachovnicový materiál
- Vytvoríme si Deformer *Collision* , ten vložíme pod tuto plochu
- Vytvoríme si objekt ktorým budeme deformovať plochu napr. guľu *Sphere*
- Deformer nastavíme v tabe *Colliders* , *Solver=Inside(Stretch)* a do zoznamu *Objects* vložíme náš objekt



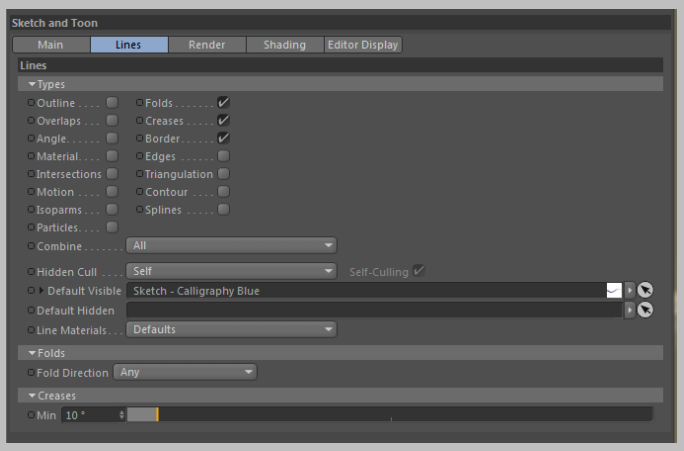
- Ako objektom pohybujeme a po ploche tak ta sa nám deformuje
- Vytvoríme si kameru a tu umiestnime presne na plochu
- Pomer strán v rendery musí byt rovnaký ako pomer strán plochy pre zvýraznenie efektu
- Kameru umiestnime do dostatočnej výšky aby zaberala celu plochu
- Po vy renderovaní získame 2D plochu ktorá simuluje deformáciu 3D objektom

- Nefotorealistické vykresľovanie sa realizuje pomocou efektu renderingu *Sketch and Toon*
- Zároveň sa tu nachádzajú aj špeciálne Shadere na tento účel
- Nastavenie efektu v renderingu môžu byt globálne ale aj nemusia, pomocou *Sketch Style tag* môžeme tieto nastavenia aplikovať len na určíte objekty
- Samotný modul Sketch and Toon je veľmi rozsiahly, takže sa budem venovať len prehľadu a použitiu, nie úprave materiálov, na toto sú v knižnici príklady a help je tiež výborný

Sketch and Toon

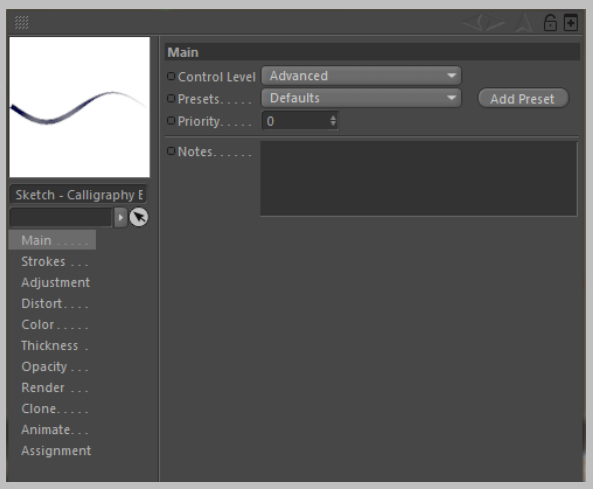


- Tento efekt je potrebne zapnúť inak ani lokálne nastavenia cez tagy sa neprejavia
- Ma 5 tabov
- Tab *Main*
  - Vyberáme si úroveň kontroly, ukážeme si len na úrovni *Simple*
- Tab *Lines*
  - Tu si vyberáme v časti *Types* čo všetko chcem aby sa kreslilo linkami
  - *Default Visible* vložím materiál ktorý chcem ako defaultný materiál liniek
  - *Default Hidden* a *Line Material s* slúžia na vyber materiálu a zapnutie kreslenia aj liniek ktoré nevidíme
  - Každý zapnutí typ so sebou prináša aj vlastne nastavenia, na tieto je najlepšie si pozrieť Help kde sú vysvetlené



- Tab *Render*
  - Tu si môžeme vybrať *Line AA* , čiže antialiasing liniek
  - A pridať objekty na ktoré nechcem aplikovať efekt *Sketch and Toon*, či objekty na ktoré len chcem tento efekt aplikovať
- Tab *Shading*
  - Tu si môžem vybrať typ pozadia, *Background* a typ zobrazenia modelu *Object*
  - Bližšie zoznámenie s módami doporučujem radšej vyskúšaním a hľadaním cez Help, kde je všetko výborné vysvetlene
  - Ak si vyberieme *Shading,* tak môžeme kvantovať farbu materiálu , čím získame ostré prechody medzi odtieňmi
  - Alebo si vytvoriť *Gradient* ktorý nám určuje ako sa bude kvantovať podľa interpolácie prechodu
- Tab *Editor Display*
  - Tu si môžeme vybrať či chceme aby nám linky zobrazovalo aj ako náhľad v editore

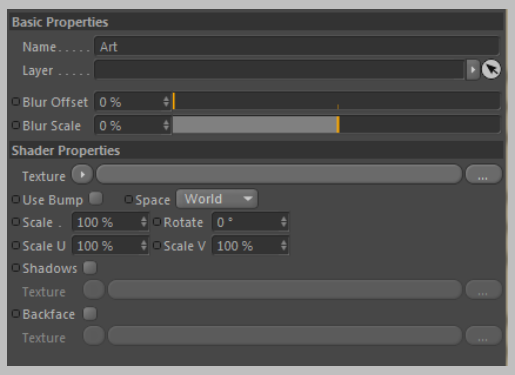
Linky



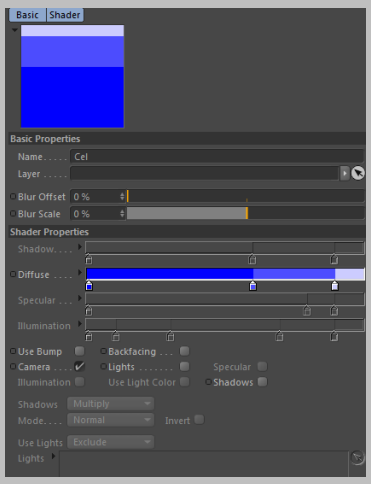
- Používajú sa na vytvoreniu efektu kreslenia liniek, na obrysy hrany či šrafovanie
- Ich použitie, čo sa nimi kresli, sa nastavuje v nastavení efektu či lokálne v tagu
- Materiál samotný je veľmi dobre modifikovateľný
- Linky nemusia byt len klasické ale môžeme simulovať aj chyby ruky
    - Ako presah linky
    - Klonovať linku pre efekt skice
- Veľa príkladov aj priamo použiteľných nájdete v knižnici Visualize-Materials-Sketch
- Je dobre si ich popozerať a uvidieť efekt rôznych nastavení priamo
- Ale aj v nastavení materiálu linky si môžeme vybrať mód zobracania nastavení, čiže na *Simple* neuvidíme pokročilé nastavenia a tak si nemusíme všimnúť ako materiál vznikol

Shadere

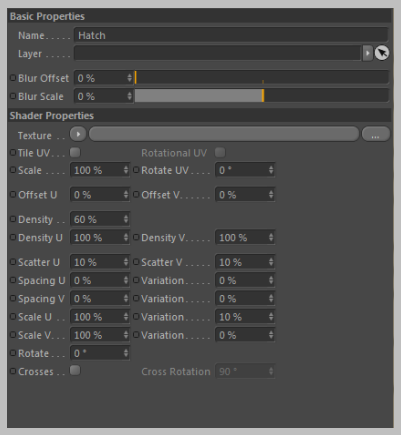
- Sketch and Toon prináša aj nove Shadere použiteľné pri tvorbe materiálov
- Na použitie týchto Shaderov netreba mat aktivovaný Sketch and Toon efekt v renderingu
- Používajú sa ako normálne Shadery
- *Art* Shader
    - Tento Shader je vytvorený na použitie v *Luminance Channel*
    - Vďaka nemu nemusíme objekt osvetľovať, ale ten sa osvetlí na základe tohto Shaderu
    - Používa sa na renderovanie ilustrácii
    - Pripomína Shadere v Zbrush
    - Tento Shader vyžaduje v sebe *Textúru,* mal by to byť obrázok gule, tuto guľu môžeme vyrenderovať či nakresliť a podľa nej sa bude vytvárať osvetlenie objektu
    - Pre pokročilé nastavenia konzultujte Help



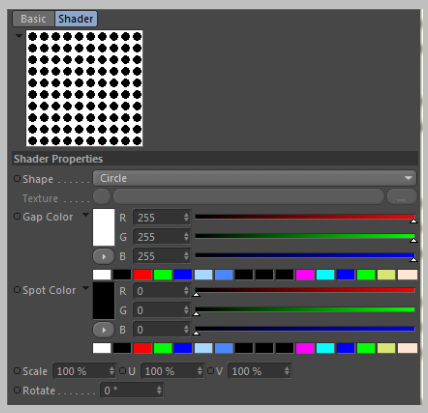
- *Cel* Shader
    - Tento Shader slúži na vytvorenie materiálu ktorý potom pri renderovaní s efektom Sketch and Toon nemusíme kvantovať aby sme získali kreslený efekt
    - Základ je v ňom si vytvoriť gradient pre *Diffuse* čo určuje farbu



- *Hatch* Shader
    - Tento Shader berie ako vstup textúru ťahu štetcom a tu potom kopíruje sa účelom dosiahnutia efektu maľby
    - Môžeme nastaviť rozloženie, medzery medzi ťahmi až po vplyv svetla na ťahy
    - Viac informácii nájdete v Help



- Spots Shader
    - Tento Shader vytvára rôzne veľké krúžky podľa svetla odrážaného do kamery
    - Týmto môžeme nasimulovať plynulý prechod z čiernej do bielej , len pomocou 2 farieb
    - Napríklad efekt tlače s nízkym rozlíšením
    - Môžeme nastaviť farbu, vplyv svetla, tieňa a pod
    - Viac v Help



Pár slov na záver

- Konzultujte help vždy ak niečo neviete
- **Pravým tlačidlom na príkaz a vybrať z ponuky *Help* je minimum**
- XPresso sa oplatí naučiť, aj keď nie vždy je z neho priamy úžitok ale urýchľuje pracú do budúcna
- Vytvárajte si vlastne knižnice
- Naučte sa MoGraph , vždy sa zíde
- Anglicka verzia je použitá preto lebo keď som sa Cinemu učil z tutoriálov ktoré boli väčšinou anglické som nechcel stále rozmýšľať ako ten či onen príkaz preložili
- Aj help je len v angličtine a tieto anglické názvy vám hľadanie v ňom omnoho urýchlia

Zdroje

Help v Cinema 4D R13

Wikipedia.com