VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

BLENDER ADD-ON PRO VYPLŇOVÁNÍ A POKRÝVÁNÍ 3D TVARŮ OBJEKTY

BLENDER ADD-ON FOR FILLING AND COVERING OF 3D SHAPES WITH OBJECTS

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **JAKUB JÚLIUS ŠMÝKAL**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. TOMÁŠ CHLUBNA**
SUPERVISOR

BRNO 2023

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

Zadání bakalářské práce

142987

Ústav: Ústav počítačové grafiky a multimédií (UPGM)
Student: **Šmýkal Jakub**
Program: Informační technologie
Specializace: Informační technologie

Název: **Blender add-on pro vyplňování a pokrývání 3D tvarů objekty**
Kategorie: Počítačová grafika
Akademický rok: 2022/23

Zadání:

1. Naučte se základy práce s 3D modelovacím programem Blender (verze 3.2 a výše).
2. Seznamte se se skriptovacím Blender Python API.
3. Navrhněte add-on, který rozšíří Blender o novou funkcionalitu vyplňování a pokrývání 3D tvarů objekty pro účely obecného generování.
4. Nastudujte možná řešení dané problematiky.
5. Navrhněte uživatelské rozhraní pro výsledný add-on.
6. Add-on implementujte a demonstrujte jeho použití na různých typech dat.
7. Zdokumentujte a zveřejněte výsledný add-on.
8. Vytvořte video reprezentující výsledky vaší práce.

Literatura:

- Blender API documentation
- Martello, Silvano, David Pisinger, and Daniele Vigo. "The three-dimensional bin packing problem." *Operations research* 48.2 (2000): 256-267.
- Christensen, Henrik I., et al. "Multidimensional bin packing and other related problems: A survey." (2016): 1-34.
- Hopper, E. B. C. H., and Brian CH Turton. "An empirical investigation of meta-heuristic and heuristic algorithms for a 2D packing problem." *European Journal of Operational Research* 128.1 (2001): 34-57.
- Freiknecht, Jonas, and Wolfgang Effelsberg. "A survey on the procedural generation of virtual worlds."
- Ebert, David S., et al. Texturing & modeling: a procedural approach. Academic Press, 2014. ISBN 1483297020, 9781483297026

Při obhajobě semestrální části projektu je požadováno:
Body 1 až 5, experimenty vedoucí k bodu 6.

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/

Vedoucí práce: **Chlubna Tomáš, Ing.**
Vedoucí ústavu: Černocký Jan, prof. Dr. Ing.
Datum zadání: 1.11.2022
Termín pro odevzdání: 10.5.2023
Datum schválení: 31.10.2022

Abstrakt

Táto práca sa zaoberá tvorbou rozšírenia do Blenderu, ktoré slúži na vypĺňanie plôch dopredu vytvorenými objektmi. Cieľom tohto rozšírenia je uľahčenie práce pri vytváraní rozsiahlych scén obsahujúcich veľké množstvo malých objektov, ktorých rozloženie je automaticky generované s určitými parametrami nastavenými užívateľom.

Abstract

This thesis is focusing on creation of Blender add-on, which can be used to fill various polygons with pre-made objects. The goal of this add-on is to simplify and speed up the process of creating large scenes with a great number of small details. The positioning of these small objects is generated with parameters specified by user.

Kľúčové slová

Blender, rozšírenie, povrch, mnohouholník, objekt, optimalizačný algoritmus, 3D modelovanie, generovanie

Keywords

Blender, add-on, surface, polygon, object, optimization algorithm, 3D modelling, generation

Citácia

ŠMÝKAL, Jakub Július. *Blender add-on pro vyplňování a pokrývání 3D tvarů objekty.* Brno, 2023. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Ing. Tomáš Chlubna

Blender add-on pro vyplňování a pokrývání 3D tvarů objekty

Prehlásenie

Prehlasujem, že som túto bakalársku prácu vypracoval samostatne pod vedením pána Tomáša Chlubny Ing. Uviedol som všetky literárne pramene, publikácie a ďalšie zdroje, z ktorých som čerpal.

.....
Jakub Július Šmýkal
5. mája 2023

Obsah

Kapitola 1

Úvod

V oblasti 3D grafiky je často potrebné vytvárať rozsiahle scény, či už pre potreby tvorby renderov, počítačových hier alebo animácii. Tieto scény väčšinou obsahujú veľké množstvo objektov, z ktorých sa môže skladať základná štruktúra scény alebo dodatočné detaily. Manuálne umiestniť všetky objekty na veľkú scénu zaberie veľa času. Z toho dôvodu je cieľom tejto práce vytvoriť rozšírenie, ktoré urýchli túto prácu a poskytne užívateľom rozšírené nastavenia rozloženia výsledných objektov pomocou náhodnosti. Rozšírenie, ktoré slúži ako všeobecný generátor scén, je možné použiť na vypĺňanie veĺkých oblastí v scénach, kde sa môže jednať napríklad o tvorbu lesov, miest alebo kamenných štruktúr.

Aj keď už existujú funkcie v Blenderi, či rozšírenia, ktoré sa tejto téme venujú, tak cieľom tejto práce je poskytnúť užívateľom alternatívu, ktorá sa zameriava na väčšiu voľnosť pri využívaní náhodnosti na pokrývanie plôch. Rozšírenie, ktoré je výsledkom tejto práce, umožňuje užívateľom meniť veľkosť a rotáciu vygenerovaných objektov. Ďalej umožňuje vytvorenie medzier medzi objektmi, prípadne prekrývanie objektov a poskytuje možnosť modifikovať geometriu objektov za účelom zahustenia výsledného pokrytia plochy. Výsledné rozšírenie je voľne dostupné pre užívateľov na stiahnutie.

Táto práca popisuje nástroje a algoritmy (kapitola 2), ktoré sú použité v implementácii rozšírenia. Ďalej popisuje návrh (kapitola 3) vytvorený za účelom tvorby nástroju, ktorý bude efektívny a jednoduchý na používanie. Práca zároveň popisuje štruktúru implementácie rozšírenia v jazyku Python s využitím rozhrania Blender API (kapitola 4). Ako posledné popisuje merania (kapitola 5), ktoré boli vykonané za účelom zistenia časovej náročnosti výsledného rozšírenia.

Kapitola 2

Teória

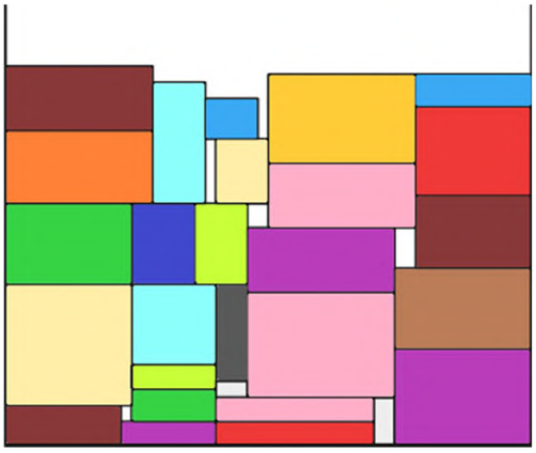
Dôležitým nástrojom pre implementáciu tohto rozšírenia pre aplikáciu Blender sú optimalizačné algoritmy na ukladanie objektov do 2D priestoru. Na ďalšie rozšírenie funkcionality výsledného programu sa využívajú pomocné nástroje ako pseudonáhodný generátor náhodných čísel a takzvané ohraničujúce objekty (Bounding Volume). Tieto ohraničujúce objekty slúžia na zjednodušnie odhalovania hraníc komplexnejších objektov a tým pádom aj na zjednodušenie implementácie optimalizačných algoritmov. Ako prostredie na vývoj a používanie rozšírenia sa používa aplikácia Blender, ktorá poskytuje aplikačné rozhranie pre prácu s jej dátovými objektmi a vstavaný textový editor spojený s prekladačom jazyka Python.

V nasledujúcich sekciách tejto kapitoly sú rozpísané rôzne metódy optimalizačných algoritmov, metódy generovania náhodných čísel, metódy ohraničovania objektov, využiteľná funkcionalita aplikácie Blender a porovnanie s už existujúcim rozšírením, ktoré sa zaoberá podobnou problematikou.

2.1 Packing Problems

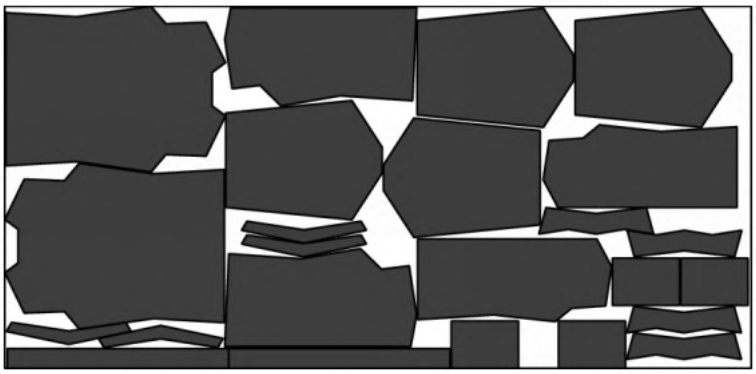
Cieľom optimalizačných algoritmov zaoberajúcich sa ukladaním objektov do kontajneru je čo najefektívnejšie vloženie objektov rôznej veľkosti a tvaru do jedného, prípadne viacerých obalov. Efektivita týchto algoritmov sa prejavuje minimalizovaním množstva nevyplneného priestoru, ktoré po dokončení behu algoritmu zostane v kontajneri. Všetky uložené objekty sa musia nachádzať celou plochou vo vnútri kontajnerov, nemôžu prekračovať jeho hranice a zároveň sa nemôžu navzájom prekrývať s ostatnými objektmi. Tieto algoritmy je možné deliť podľa počtu dimenzií, ktoré sa musia zohľadňovať pri ukladaní objektov [15]. Táto práca sa zameriava na ich využitie v 2D priestore.

Problém balenia objektov v 2D priestore (anglicky 2D packing problem) je typ kombinatorických optimalizačných problémov, ktorých cieľom je vložiť objekty na plochu určitého tvaru. V reálnom živote sa táto optimalizácia využíva pri výrobe, kedy je potrebné minimalizovať nevyužitý materiál pri vyrezávaní tvarov do určitého materiálu. V informatike sa problém balenia objektov v 2D priestore považuje za NP problém, ktorý je väčšinou riešený pomocou heuristických algoritmov. Podľa pravidelnosti tvarov ukladaných objektov je možné problémy ukladania objektov rozdeliť na 2D pravidelné a 2D nepravidelné. V prípade 2D pravidelných problémov (obrázok 2.1) sa všetky objekty problému skladajú z pravidelných mnohouholníkov, ako napríklad obdĺžniky alebo kruhy. Pri 2D nepravidelných problémoch (obrázok 2.2) sa môžu objekty skladať z rôznych zložitých konkávnych tvarov [11].



Obr. 2.1: Ukladanie pravidelných objektov v 2D priestore do pravidelného kontajneru bez prekrývania jednotlivých objektov. Hrany všetkých vkladaných objektov sú paralelné s hranami kontajneru[2].

Obtiažnosť týchto algoritmov závisí od tvaru jednotlivých objektov a kontajnerov, do ktorých sa ukladajú. Najjednoduchšie problémy balenia objektov sú tie, pri ktorých majú všetky objekty a kontajnery tvar obdĺžniku. Z toho dôvodu sa veľké množstvo vedeckých prác, ktoré sa zaoberajú problémom ukladania objektov v 2D alebo 3D priestore zameriavajú skôr na implementáciu s objektmi jednoduchých tvarov. Avšak pri väčšine praktických využitiach majú objekty a kontajnery nepravidelné tvary. Kvôli geometrickej zložitosti, ktorú prinášajú nepravidelné tvary, nie je tento typ problémov ukladania objektov až tak dobre preskúmaný ako ukladanie objektov jednoduchých tvarov [5].



Obr. 2.2: Ukladanie nepravidelných objektov v 2D priestore do pravidelného kontajneru bez prekrývania jednotlivých objektov[4].

Na zjednodušenie riešenia problémov ukladania nepravidelných objektov v 2D priestore sa používa metóda obaľovania mnohouholníkom. Táto metóda je založená na princípe, kedy je objekt nepravidelného tvaru obalený to takzvanej obálky, ktorá má pravidelný tvar, ako napríklad obdĺžnik, kruh, trojuholník alebo iný pravidelný mnohouholník. Do kontajneru

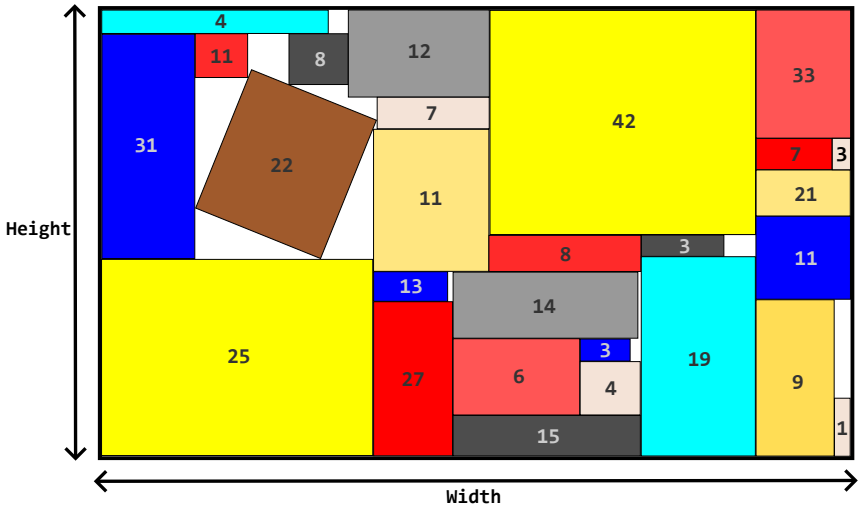
[2] https://www.frontiersin.org/articles/10.3389/fmech.2022.966691/full
[4] https://www.hindawi.com/journals/mpe/2014/548957/

sú potom ukladané už obalené objekty a preto je možné k problému pristupovať ako k problému ukladania pravidelných objektov. Táto metóda výrazne zjednodušuje ukladanie objektov do kontajneru, avšak zároveň znižuje efektivitu riešenia. Obálky, pomocou ktorých sú nepravidelné objekty obaľované znižujú výslednú hustotu objektov v kontajneri a tým prispievajú k zväčšeniu nevyužitej plochy [11].

2.1.1 2D Knapsack Problem

Tento problém (obrázok 2.3) sa zameriava na ukladanie menších objektov s rôznou hodnotou do jedného väčšieho kontajneru. Cieľom nie je vložiť do kontajneru všetky objekty, ale vložiť takú podmnožinu objektov, pri ktorej bude súčet hodnôt jej objektov ten najvyšší, ktorý je z danej skupiny objektov možné vložiť do kontajneru [1].

Všetky tieto objekty aj s kontajnerom majú tvar obdĺžniku, ktorý je definovaný jeho rozmermi. V 2D priestore je definovaný jeho výškou a šírkou. Jednotlivé objekty sú zároveň definované hodnotou, ktorá sa používa ako miera kvality vyplnenia kontajneru. Ďalej sú definované množstvom kópii, ktoré existujú v rámci daného riešenia problému a je možné ich využiť počas ukladania [3]. V prípade, že hodnota objektu je závislá na ploche, ktorú daný objekt pokrýva, tak je cieľom tohto problému vytvoriť uloženie pokrývajúce čo najväčšiu plochu kontajneru.



Obr. 2.3: 2D Knapsack Problem pre pravidelné objekty. Všetky vkladané objekty sú definované ich rozmermi a hodnotou. Rozmery určujú plochu, ktorý objekty pokryjú v kontajneri. Hodnota, zobrazená pomocou čísel nachádzajúcich sa na jednotlivých objektov, sa používa na určenie kvality vyplnenia.

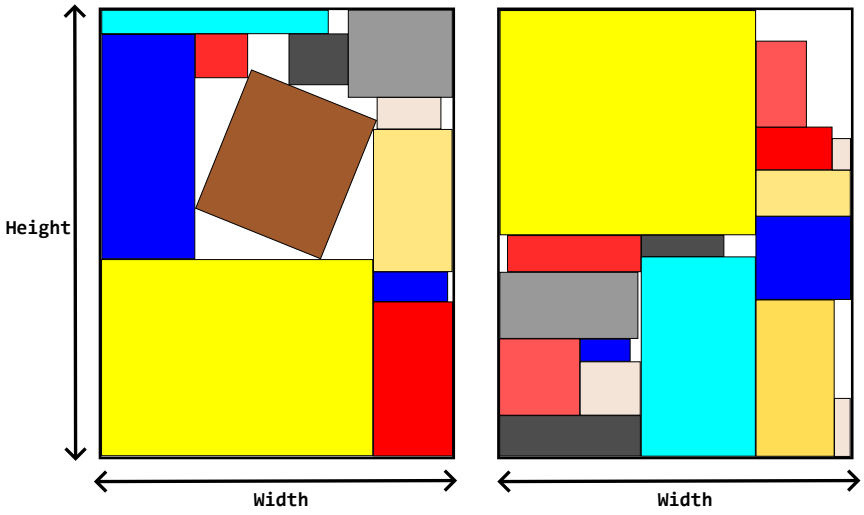
2.1.2 2D Bin Packing Problem

V prípade, že sa pracuje s pravidelnými objektmi nachádzajúcimi sa v 2D priestore, je tento problém (obrázok 2.4) v informatike považovaný za kombinatorický NP-ťažký problém. Jeho cieľom je vložiť konečný počet objektov do minimálneho počtu kontajnerov bez toho, aby sa tieto objekty prekrývali. Zároveň tieto objekty nemôžu presahovať cez hranice kontajneru. Ako objekty sa zvyčajne používajú pravidelné tvary, ako obdĺžnik, prípadne štvorec.

Kontajnery majú zvyčajne tiež tvar obdĺžniku. V tomto type problému, kedy sa jedná o objekty 2D priestoru, sú všetky objekty a kontajnery definované pomocou dvoch rozmerov - výšky a šírky. Tieto rozmery sa môžu v prípade objektov líšiť. Avšak pre kontajnery je potrebné, aby mali všetky jednotnú výšku a šírku [6].

Výsledkom algoritmov, ktoré riešia bin packing problém, je uloženie objektov v minimálnom počte kontajnerov. Zároveň sa tieto algoritmy snažia o čo najefektívnejšie využitie priestoru dostupného v každom kontajneri, aby sa minimalizovala nevyužitá plocha.

V reálnom svete má tento problém využitie vo všetkých priemysloch, ktoré pri svojej produkcí potrebujú vyrezávať rôzne tvary z materiálu. Tento materiál má zvyčajne tvar obdĺžniku alebo štvorcu. Z toho dôvodu je potrebné z každého kusu materiálu vyrezať čo najväčšie množstvo objektov, nech sa minimalizuje množstvo vytvoreného odpadu [6].



Obr. 2.4: 2D Bin Packing Problem pre pravidelné objekty, ktoré sú definované ich rozmermi. Tieto objekty môžu byť vkladané do viacerých kontajnerov. Všetky kontajnery musia mať rovnaké rozmery.

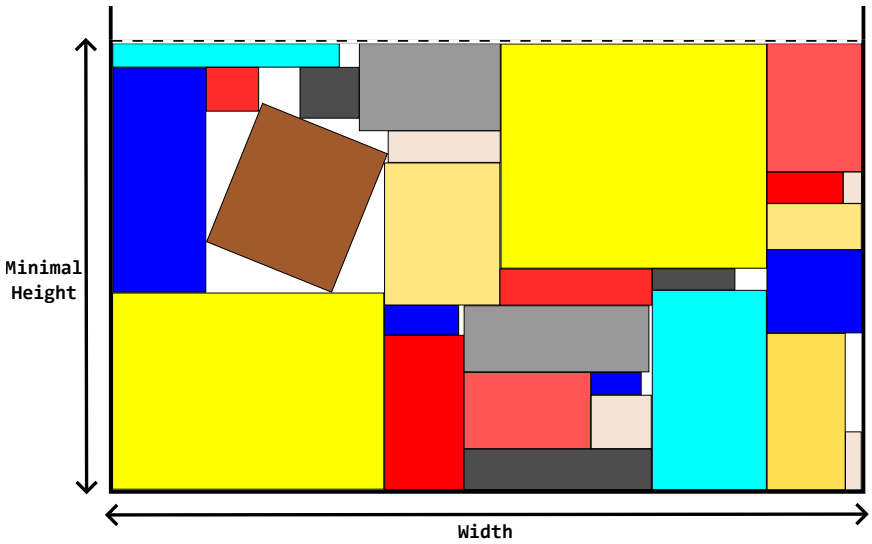
2.1.3 2D Strip Packing Problem

Pri pravidelných objektoch nachádzajúcich sa v 2D priestore sa tento problém (obrázok 2.5) v informatike považuje za problém s komplexnosťou NP-ťažký. Tento problém sa zameriava na vloženie konečného počtu objektov obdĺžnikového tvaru do jedného kontajneru. Tento kontajner má tiež tvar obdĺžniku, avšak má pevne definovaný len jeden rozmer, svoju šírku, ale jeho výška je neobmedzená [7]. Jednotlivé objekty sú definované pomocou ich šírky a výšky, ktoré nemusia byť pre všetky objekty rovnaké.

Riešením tohto problému je minimálna výška potrebná pre vloženie všetkých objektov do kontajneru s ohľadom na určité obmedzenia. Všetky objekty sú do kontajneru vložené ortogonálne bez toho, aby sa navzájom prekrývali, alebo aby presahovali hranice kontajneru, ktoré sú v tomto prípade definované iba šírkou kontajneru.

Tento problém je možné využiť v priemysle, kde sa vyrezávajú tvary z materiálu, ktorý má tvar obdĺžniku. Avšak pri tomto probléme sa musí jednať o dostatočne veľkú plochu materiálu, aby bolo možné jej výšku považovať za nekonečnú. Jeho ďalšie využitie je pri

problematike plánovania úloh, kde jednotlivé úlohy vyžadujú súvislú časť určitého zdroju. V tomto prípade sa napríklad za zdroj môže považovať čas a za objekt čas potrebný na dokončenie danej úlohy [7].



Obr. 2.5: 2D Strip Packing Problem pre pravidelné objekty definované ich rozmermi. Kontajner je definovaný len jedným rozmerom - šírkou. Riešením tohto problému je minimálna výška potrebná na uloženie všetkých dostupných objektov.

2.1.4 2D Orthogonal Packing Problem

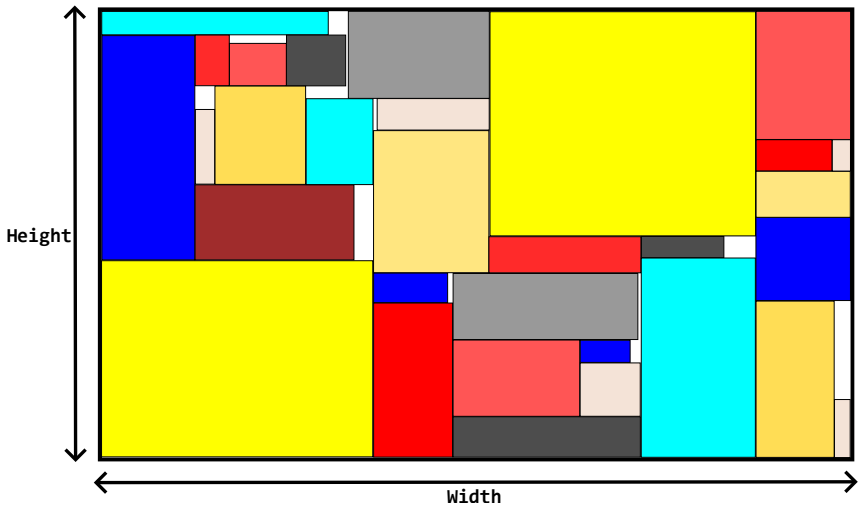
Tento problém (obrázok 2.6) sa pri pravidelných objektoch považuje jeho zložitosťou za NP-complete. Zameriava sa na ukladanie konečného množstva ortogonálnych objektov do konečného množstva ortogonálnych kontajnerov. Všetky objekty a kontajnery sú definované pomocou ich rozmerov, v prípade 2D priestoru ich výškou a šírkou.

Cieľom tohto problému je vložiť všetky objekty do minimálneho množstva kontajnerov tak, aby boli splnené základné podmienky tohto problému. Tieto podmienky diktujú, že jednotlivé objekty sa nesmú prekrývať a zároveň nesmú presahovať za hranice kontajneru. Ďalej musia byť tieto objekty uložené takým spôsobom, aby všetky ich hrany boli paralelné s hranami kontajneru, v ktorom sa nachádzajú [4].

Tento problém má rovnako ako aj ostatné problémy tohto typu využitie v priemysle, kde sa z materiálu obdĺžnikového tvaru vyrezávajú menšie objekty rovnakého tvaru. Ďalej je možné tento problém využiť pri tvorbe návrhu rozloženia stránok novín alebo webových stránok [8].

2.2 Pseudonáhodné generovanie čísel

Existujú dva rôzne spôsoby, ktorými je možné generovať náhodné čísla. Prvý spôsob je založený na implementácií pomocou hardware. Ten je považovaný za pravý generátor náhodných čísel. Ďalší spôsob je generovanie čísel založené na implementácii pomocou software, ktoré sa považuje za pseudonáhodné generovanie čísel (anglicky Pseudo Random Number Generator, ďalej len PRNG). Pravý generátor náhodných čísel používa na generovanie dáta získané z



Obr. 2.6: 2D Orthogonal Packing Problem pre pravidelné objekty, ktoré sú definované ich rozmermi. Všetky vkladané objekty musia mať hrany paralelné s hranami kontajneru.

prírodných fenoménov, ako napríklad atmosférický alebo termálny hluk. Tieto postupnosti čísel majú lepšiu kvalitu, než náhodné čísla generované pomocou software. Avšak typické užívateľské zariadenia neobsahujú technológie, pomocou ktorých by bolo možné sledovať a merať prírodné fenomény s dostatočnou presnosťou pre potreby generátoru. Preto sa na generovanie náhodných čísel bežne využívajú skôr PRNG [13].

Pod pojmom PRNG sa označujú algoritmy, ktoré využívajú matematické vzorce, aby vytvorili postupnosť pseudonáhodných čísel. Tieto postupnosti ponúkajú približne podobné vlastnosti ako pravé náhodne generované čísla. Na spustenie PRNG sa využíva hodnota seed, pomocou ktorej sa inicializuje PRNG do počiatočného stavu [17], z ktorého potom algoritmus vytvára veľké množstvo čísel v relatívne krátkom čase.

Typická štruktúra PRNG sa skladá z konečného množstva stavov, výstupnej hodnoty a dvoch funkcií, z ktorých jedna slúži na prechod medzi stavmi (rovnica 2.1) a druhá na vypočítanie výstupnej hodnoty (rovnica 2.2).

$$f : S_{n-1} \to S_n \tag{2.1}$$

S predstavuje jednotlivé stavy PRNG

$$g : S_n \to U_n \tag{2.2}$$

S predstavuje stav PRNG, U predstavuje výstupnú hodnotu PRNG

Pri spúšťaní PRNG je zvyčajne možné si zvoliť rozsah, v ktorom sa vygenerované čísla budú nachádzať. Ďalej sa pri jeho spúšťaní nastavuje hodnota seed, ktorá slúži ako počiatočná hodnota v stave S_0. Po inicializovaní sa začne generovanie sekvencie čísel (rovnice 2.3 a 2.4).

$$S_n = f(S_{n-1}) \tag{2.3}$$

$$U_n = g(S_n) \tag{2.4}$$

S_n predstavuje stav číslo n pri generovaní sekvencie pomocou PRNG
U_n predstavuje výstupnú hodnotu číslo n pri generovaní sekvencie pomocou PRNG

Pri týchto rovniciach je možné si všimnúť, že skôr než k náhodnému generovaniu dochádza k výpočtom podľa určitej postupnosti. Vďaka tomuto je možné dosiahnúť toho, že pri viacerých spusteniach PRNG s tou istou hodnotou seed sa vygeneruje vždy tá istá postupnosť čísel. Aby sa vygenerovala odlišná postupnosť je potrebné PRNG inicializovať s inou hodnotou seed.

PRNG má obmedzené miesto na jednotlivé stavy a preto sa časom pri generovaní čísel dostane späť do určitého stavu, v ktorom už pri danom behu bol. To spôsobuje, že sa sekvencia vygenerovaných čísel začne opakovať. Najmenšie možné číslo p, pre ktoré platí, že PRNG musí prejsť p počtom iterácii, aby sa vrátil do rovnakého stavu, sa nazýva perióda generátoru. Čím je táto perióda dlhšia, tým sa hodnoty v sekvencií vygenerovanej pomocou PRNG menej často opakujú. Vďaka tomu sa sekvencia približuje viac k sekvencií vytvorenej pomocou pravého generátoru náhodných čísel. Preto je potrebné si zvoliť takú dĺžku periódy, aby PRNG vytváralo dostatočne náhodnú postupnosť pre potreby užívateľa. Rozlišujeme viacero typov PRNG, ako napríklad lineárny kongruentný generátor (rovnica 2.5), viacnásobný rekurzívny generátor alebo rôzne kombinované generátory [12].

$$x_n = (a \cdot x_{n-1} + c)\ mod\ m \tag{2.5}$$

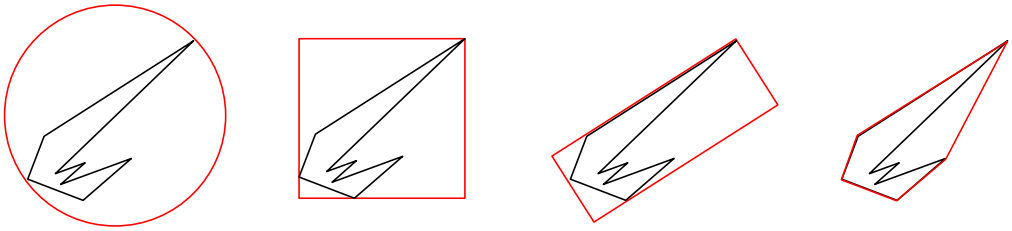
Príklad lineárneho kongruentného generátoru náhodných čísel.
x predstavuje stav, x_0 predstavuje počiatočný stav (seed), a je lineárny násobiteľ, c je inkrement a m určuje rozsah vygenerovaného čísla $< 0,\ m - 1 >$ [14]

PRNG sa využíva v aplikáciách, kde je potrebné mať väčšie množstvo náhodne vygenerovaných čísel a je vhodné, aby bolo možné túto postupnosť vygenerovať viac krát. To môžu byť napríklad aplikácie vytvorené za účelom modelovania určitého systému alebo vytvárania simulácii. Na druhú stranu obyčajné PRNG nie je vhodné v aplikáciách, ktoré potrebujú aby vygenerované čísla boli skutočne nepredvídateľné, ako napríklad rôzne systémy na šifrovanie dát. V tomto prípade sa používajú čo najviac nepredvídateľné PRNG. Za tieto môžeme považovať PRNG využívajúce na svoju inicializáciu hodnoty, ktoré sú ťažko replikovateľné, ako napríklad dáta z jednotlivých hardvérových komponentov a periférii počítaču [16].

2.3 Bounding Volume

Za účelom zjednodušovania simulácií kolízii v 2D a 3D priestore sa využíva ohraničenie objektov menej komplexnými tvarmi. Toto ohraničenie sa nazýva Bounding Volume. Táto metóda urýchľuje odhalovanie kolízii tým spôsobom, že nahradzuje potrebu kontroly kolízie zložitých objektov kontrolov iba ich zjednodušených obalov [10]. Ako ohraničenie sa zvyčajne používajú najmenšie geometrické primitíva, ktoré obaľujú celý objekt. Medzi tieto

geometrické objekty používané v 3D priestore patrí napríklad kváder (bounding box), guľa (bounding sphere) alebo zjednodušený objekt tvorený čisto konvexnými polygónmi (convex hull). V 2D priestore sa používajú 2D ekvivalenty týchto objektov (obrázok 2.7). Pri použití tých najjednoduchších tvarov ako napríklad kvádrov alebo obdĺžnikov sa ďalej môžu obaly deliť podľa toho, či dedia rotácia objektu, ktorý ohraničujú alebo dedia rotáciu priestoru, v ktorom sa nachádzajú [9].



Obr. 2.7: Príklady ohraničujúcich objektov vytvorených v 2D priestore pre konkávny objekt. Medzi príkladmi sa nachádza ohraničenie pomocou kruhu (Bounding Sphere), ohraničenie štvorcom/obdĺžnikom s rotáciou definovanou osami 2D priestoru (Axis-Aligned Bounding Box), ohraničenie štvorcom/obdĺžnikom s rotáciou zdedenou od objektu (Oriented Bounding Box) a ohraničenie zjednodušenou aproximáciou objektu pomocou konvexného tvaru (Convex Hull)[6].

2.3.1 Bounding Sphere

Tento spôsob je založený na ohraničení objektu pomocou gule. Toto ohraničenie je definované pomocou stredového bodu a polomeru gule. V 2D priestore sa na ohraničenie používa kruh. Hlavná výhoda tohto ohraničenia je jeho nemennosť v závislosti na rotácií objektu, takže ohraničenie si vždy ponecháva svoju pôvodnú rotáciu. Tento typ ohraničenia je vhodný iba v situácií, kedy objekt ktorý ohraničuje má guľovitý tvar. V opačnom prípade zaberá zbytočne veľa miesta a je vhodné ho nahradiť iným typom [10].

2.3.2 Axis-Aligned Bounding Box

Pri tomto spôsobe sa ohraničenie tvorí pomocou kvádra, ktorý nededí rotáciu od objektu, ale od scény. Rotácia 3D scény je založená na troch osách X,Y a Z. Jeho hlavná výhodou je výpočtová jednoduchosť pri simuláciách a jednoduché nájdenie kvádra pre zložitý objekt. V 2D priestore je ohraničenie tvorené obdĺžnikom. Keďže nededí rotáciu od objektu, tak v prípade otočenia objektu sa prepočítavajú rozmery jeho ohraničenia. Tento typ na rozdiel od orientovaného kvádra obsadzuje viac nevyplneného priestoru [9].

2.3.3 Oriented Bounding Box

Objekt je ohraničený do kvádra, ktorý pri použití tohto spôsobu dedí rotáciu obalovaného objektu. Tento kváder je definovaný pomocou ôsmich bodov rozložených v priestore. V 2D priestore sa používa obdĺžnik. Rotácia tohto typu ohraničenia je závislá na rotácií objektu, čo spôsobuje že pri rotácii objektu sa nemusí prepočítavať ohraničenie a zbytočne obsadené miesto bude minimálne. Avšak pri simuláciách je použitie orientovaného kvádru výpočtovo náročnejšie, než použitie kvádru, ktorý nededí rotáciu objektu [9].

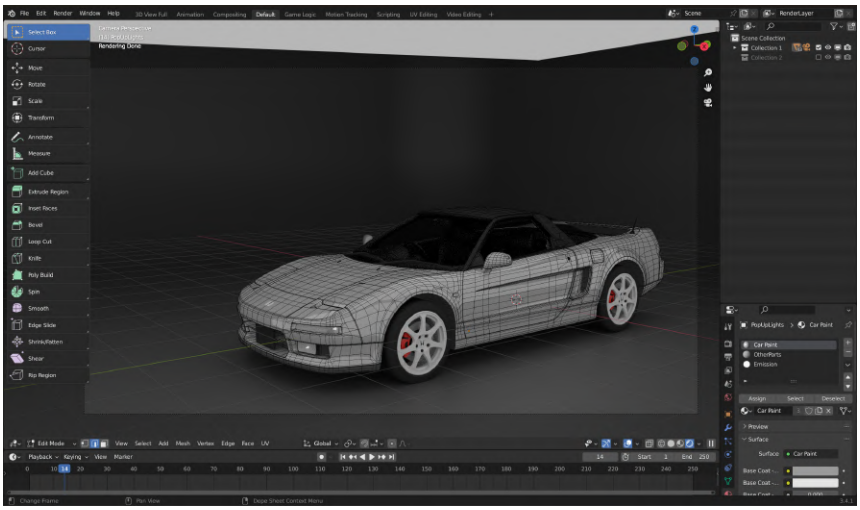
[6] https://ncollide.org/bounding_volumes/

2.3.4 Convex Hull

Tento spôsob je založený na použití tvaru objektu na vytvorenie jeho ohraničenia. Toto ohraničenie je konvexný objekt, ktorý je zjednodušenou aproximáciou pôvodného objektu. Keďže kopíruje hlavné tvary objektu tak sa jedná o veľmi presné ohraničenie, ktoré obsadzuje minimum nadbytočného miesta. Avšak z dôvodu, že nie je tvorené iba jediným geometrickým primitívom je toto ohraničenie výpočetne náročné. V 2D priestore sa používa najmenší konvexný polygón, do ktorého sa zmestia všetky body objektu [9]. Tento spôsob sa používa v prípade, že geometrické primitíva sú príliš nepresné pre potreby simulácie alebo detekcie kolízie.

2.4 Blender

Blender je open-source nástroj určený na vytváranie a prácu s 3D objektmi, scénami a animáciami (obrázok 2.8). Cieľom tejto aplikácie je sprístupniť pokročilé technológie na tvorbu 3D užívateľom zdarma. Preto od roku 2002 je Blender voľne dostupný na internete pod licenciou GPL[7]. Blender podporuje všetky základné nástroje určené na tvorbu v 3D, ako napríklad modelovanie, tvorba textúr, mapovanie objektov na kostry, tvorbu animácii, tvorbu simulácií, sledovanie pohybu a aj úpravu videa. Kedysi bol priamo v aplikácií dostupný aj engine určený na tvorbu hier, ale podpora jeho vývoju skončila v roku 2019.



Obr. 2.8: Scéna vytvorená pomocou aplikácie Blender

Okrem 3D tvorby umožňuje Blender užívateľom vytvárať rôzne rozšírenia, ktoré slúžia na uľahčenie práce alebo na celkové rozšírenie funkcionality aplikácie. Užívatelia môžu do Blenderu vytvoriť rady nových nástrojov alebo upraviť tie pôvodné. Na to slúži aplikačné rozhranie Blender API, ktoré umožňuje interakciu medzi Blenderom a jazykom Python.

2.4.1 Blender API

Blender poskytuje užívateľom aplikačné rozhranie Blender API, ktoré slúži na modifikovanie jeho funkcionality. K tomuto účelu má Blender v sebe zabudovaný interpret pre jazyk

[7] https://www.blender.org/about/

Python, ktorý sa automaticky načíta pri spustení Blenderu a je dostupný počas jeho celého behu. Tento prekladač poskytuje užívateľovi Blender moduly pre Python, ako `bpy` a `mathutils`. Tieto moduly umožňujú užívateľovi pristupovať k dátam, triedam a funkciám, ktoré sa v Blenderi nachádzajú [2].

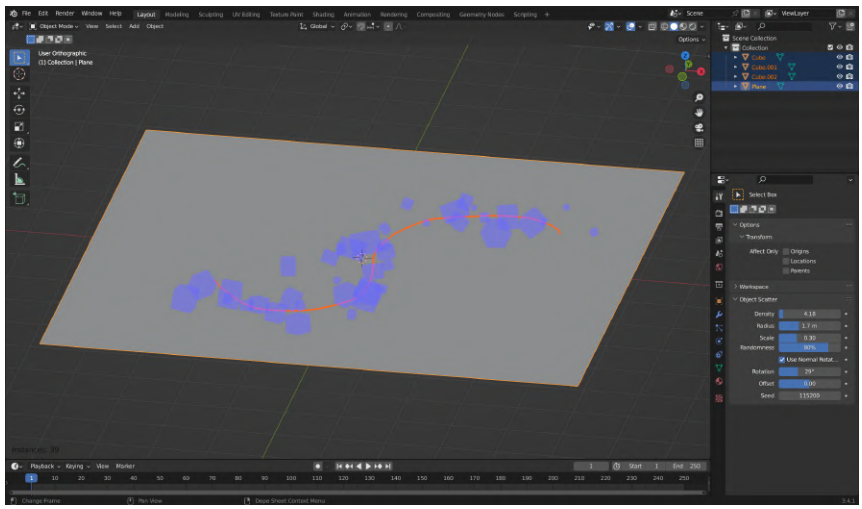
Všetky skripty, ktorých cieľom je pracovať s Blenderom a upravovať jeho funkcionalitu potrebujú k správnemu fungovaniu tieto moduly. Typické využitie týchto skriptov je rozšírenie užívateľského prostredia, manipulácia so scénou a objektmi, automatizácia pracovného postupu a vytváranie vlastných nástrojov. Tieto skripty potom môžu byť vložené do Blenderu ako rozšírenia.

2.4.2 Rozšírenia

Blender umožňuje užívateľom upraviť svoju funkcionalitu inštaláciou rozšírení. Tieto rozšírenia je možné inštalovať cez preferencie. V základe Blender prichádza s množstvom už predinštalovaných rozšírení, ktoré je potrebné len podľa potreby užívateľa povoliť. Tieto rozšírenia spadajú do dvoch kategórii. Podľa toho, kto je ich autorom sa rozdeľujú na oficiálne a komunitné. Avšak užívateľ môže nainštalovať svoje vlastné rozšírenia. Prípadne môže nainštalovať iné rozšírenia, ktoré získal ich stiahnutím z internetu [2].

2.4.3 Scatter Objects

Jeden z nástrojov, ktorý je predinštalovaný v Blenderi a zaoberá sa podobnou problematikou ako táto práca sa nazýva Scatter Objects. Tento nástroj pridáva do Blenderu funkcionalitu pokrývania plochy inými objektmi, avšak pracuje na odlišnom princípe než výsledok tejto práce.



Obr. 2.9: Rozšírenie Scatter Objects, zaoberajúce sa podobnou témou ako táto práca

Rozšírenie Scatter Objects nevypĺňa plochu objektmi po polygónoch, ale pomocou čiary, ktorú nakreslí užívateľ. Objekty sú vytvorené iba v okolí tejto čiary a nie po celej ploche. Poloha a rotácia vytvorených objektov je nastavená pomocou užívateľom stanovených parametrov (obrázok 2.9). Tým pádom, aj keď sa toto rozšírenie zaoberá vypĺňaním plochy objektmi, tak sa k tejto problematike stavia z iného uhlu pohľadu než táto práca, ktorej cieľom je čo najoptimálnejšie vyplniť celé plochy. Toto rozšírenie síce ponúka zaujímavý

pohľad na túto problematiku, avšak nie je možné ho priamo porovnávať s výsledkom tejto práce. Ale na druhú stranu je možné preskúmať spôsob, akým užívateľ interaguje s týmto rozšírením, odhaliť jeho nedostatky a využiť tieto vedomosti pri zlepšení výsledného produktu tejto práce.

Pri používaní tohto rozšírenia je možné si povšimnúť, že ovládanie nie je úplne užívateľsky prívetivé. Rozšírenie obsahuje užívateľské rozhranie, ale to sa používa iba na zmenu parametrov, nevyužíva sa na základné ovládanie rozšírenia. Objekty sa vyberajú v 3D scéne, kedy všetky vybraté objekty, až na posledný, sa použijú na pokrytie. Posledný vybratý objekt je pokrývaný. Po potvrdení vytvorenia objektov už nie je možné tieto objekty upravovať pomocou parametrov v paneli vlastností. V prípade potreby väčšieho zásahu do vytvorených objektov je potrebné nakresliť čiaru znovu. Zároveň nie je možné pristupovať ku geometrií týchto objektov priamo, ale iba cez jeden objekt - ich rodičovský objekt, pri ktorého upravení sa vykonajú zmeny na všetkých týchto objektoch. Cieľom tejto práce je vytvoriť vypĺňanie plochy objektmi z iného pohľadu, ako toto rozšírenie a zároveň sa vyhnúť nedostatkom, ktoré sa v tomto rozšírení nachádzajú.

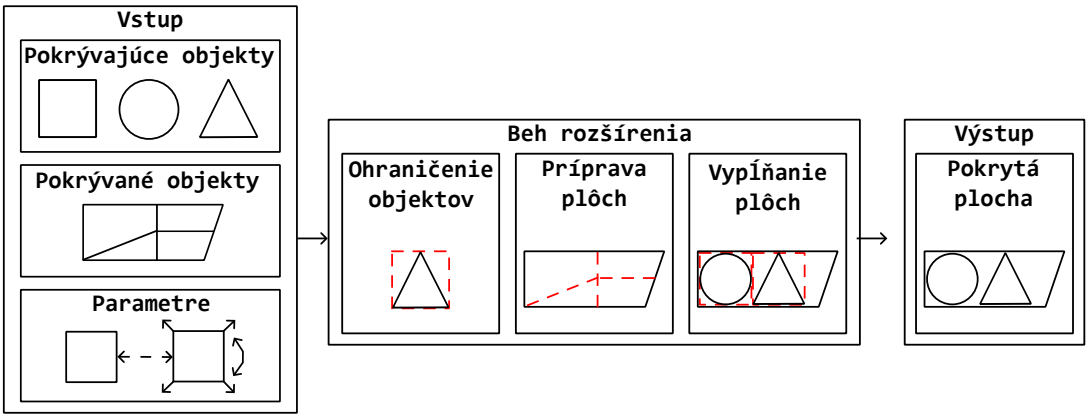
Kapitola 3

Návrh

V nasledujúcich sekciách tejto kapitoly je rozpísaný návrh rozšírenia popisujúci jeho fungovanie, prácu s ním pomocou užívateľského prostredia, jeho vstup a výstup.

3.1 Popis rozšírenia

Cieľom tejto práce je vytvoriť obecný generátor pre aplikáciu Blender určený na tvorbu rozsiahlych repetitívnych detailov v scénach. Tento generátor je založený na pokrývaní plôch pomocou užívateľom vytvorených objektov. Objekty určené na pokrývanie sú vyberané zo zoznamu objektov náhodne. Ich uloženie na plochu je založené na základe optimalizačných algoritmov zameraných na ukladanie pravidelných objektov do pravidelných kontajnerov.



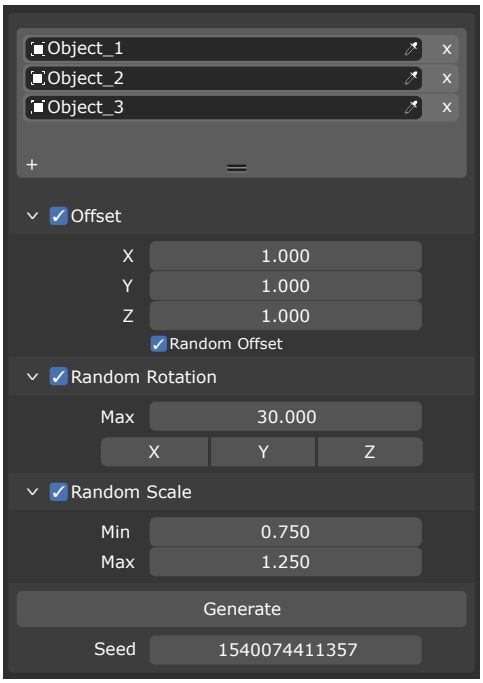
Obr. 3.1: Obrázok zobrazuje vstupy, ktoré rozšírenie požaduje, priebeh behu rozšírenia a výstup, ktorý jeho behom vzniká. Beh rozšírenia sa skladá s viacerých častí - prípravy pokrývajúcich objektov, kedy sa tieto objekty obalia do ohraničenia, prípravy pokrývaných plôch, kedy sa zjednoduší ich geometrie a vypĺňanie tejto zjednodušenej plochy pomocou objektov.

Vstupom rozšírenia sú objekty určené na pokrývanie plôch, objekty, ktorých plochy budú pokrývané a užívateľom nastavené parametre (obrázok 3.1). Po dokončení behu tohto rozšírenia vznikne množina objektov, ktoré budú v 3D scéne pokrývať plochy všetkých užívateľom zvolených objektov . Toto rozšírenie bude možné použiť pri pokrývaní veľkých plôch v scénach, pri ktorých je potrebná určitá náhodnosť, ale nie dostatočná na to, aby si vyžadovala manuálne ukladanie objektov. Príklad takéhoto použitia je rozloženie stromov,

kríkov a kameňov v prírode. Zároveň bude možné pomocou rozšírenia doplniť do scén veľké množstvo malých detailov, ako napríklad kamienky na dne rieky alebo náhodne rozptýlené nástroje a nádoby na záhrade.

3.2 Užívateľské prostredie

Ovládanie rozšírenia je zabezpečené pomocou užívateľského rozhrania. Pre uľahčenie práce s nástrojom bolo cieľom tejto práce vytvoriť čo najjednoduchšie a najintuitívnejšie užívateľské prostredie. Kládol sa dôraz na to, aby bolo možné s rozšírením pracovať aj bez toho, aby bol užívateľ nútený čítať dokumentáciu. V prípade vzniku menších nejasností každý parameter a operátor v užívateľskom rozhraní obsahuje krátky popis jeho funkcionality. Návrh užívateľského rozhrania (obrázok 3.2) bol vytvorený pomocou vektorovej grafiky.



Obr. 3.2: Návrh užívateľského prostredia zobrazujúci vyberanie objektov určených na pokrývanie plôch a užívateľské parametre. Základné ovládanie rozšírenia sa nachádza v hlavnom paneli užívateľského prostredia. Voliteľné parametre sú rozdelené do viacerých podpanelov, ktoré je možné v prípade potreby schovať. Na spustenie rozšírenia slúži operátor Generate, pod ktorým sa ešte nachádza hodnota seed, využívaná na inicializáciu pseudonáhodného generátoru čísel.

Na zabezpečenie čo najväčšej jednoduchosti používania boli v užívateľskom prostredí ponechané len tie najzákladnejšie atribúty, ktoré sú potrebné pre prácu s týmto rozšírením. Po nastavení atribútov sa spustí funkcionalita tohto rozšírenia pomocou operátoru `Generate`. V prípade, že nebudú splnené základné predpoklady pre jeho spustenie, bude tento operátor nedostupný, o čom bude užívateľ informovaný jeho zašednutím.

Užívateľské prostredie sa skladá z hlavného panelu, na ktorom sa nachádzajú všetky atribúty potrebné na jeho spustenie a operátor, ktorý ho spustí. Ďalej sa skladá z troch podpanelov. Každý z nich obsahuje voliteľné parametre. Tieto parametre je možné deaktivo-

vať pomocou checkboxu, ktorý sa nachádza v záhlaví každého tohto podpanelu. V prípade, že je parameter deaktivovaný, všetky jeho nastavenia sú zašednuté a nie je možné ich meniť.

3.3 Parametre rozšírenia

Užívateľské prostredie rozšírenia umožňuje užívateľovi ovplyvniť výsledok behu rozšírenia pomocou nastavenia vstupných parametrov. Tieto parametre sú rozdelené do dvoch skupín podľa toho, či sú pre beh rozšírenia potrebné.

3.3.1 Povinné parametre

Tieto parametre je potrebné nastaviť, aby bolo možné rozšírenie použiť. Patrí k nim objekt, prípadne objekty, ktoré budú použité na vypĺňanie a minimálne jeden objekt, ktorého plochy budú týmito objektmi pokryté. Ako posledné k ním patrí hodnota `seed`.

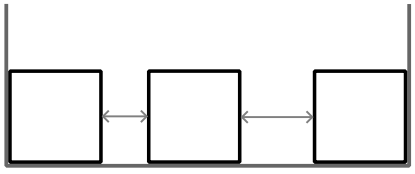
Za objekty, ktorých plochy budú pokryté sa považujú všetky objekty, ktoré sú v 3D priestore označené. Objekty, ktoré budú použité na vypĺňanie sa vyberajú pomocou užívateľského prostredia, kde je možné ich vybrať buď zo zoznamu všetkých existujúcich objektov alebo pomocou takzvaného pickeru, ktorý do zoznamu vloží užívateľom vybratý objekt z 3D priestoru. Počet objektov využívaných na vypĺňanie plochy je neobmedzený. Komplexnosť geometrie a počet objektov, ktoré budú pokrývané tiež nie je obmedzený, ale geometria týchto objektov bude mierne pred ich vyplnením zjednodušená. Cieľom týchto úprav je urýchlenie behu operátoru a zníženie počtu malých, zložitých polygónov, ktoré by nebolo možné pokryť vybratými objektmi.

Posledným povinným parametrom je `seed`, ktorý slúži na inicializovanie pseudonáhodného generátoru čísel. Tento generátor sa používa na výber zo zoznamu objektov určených na pokrývanie plôch. Vďaka tomuto parametru je možné dosiahnuť toho istého výsledku pri viac násobnom použití operátoru s rovnako nastavenými ostatnými parametrami rozšírenia.

3.3.2 Voliteľné parametre

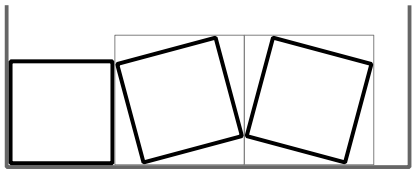
V rozšírení sa nachádzajú tri vstupné voliteľné parametre - `Offset`, `Random Rotation` a `Random Scale`. Vďaka týmto parametrom je možné viac ovplyvniť výsledky tohto rozšírenia, čím sa zlepšuje jeho užitočnosť pre používateľov.

Parameter `Offset` (obrázok 3.3) pridáva možnosť vytvoriť medzery medzi jednotlivými vygenerovanými objektmi. Veľkosť tejto medzery je možné nastaviť v smere každej osi zvlášť. Zároveň je možné nastaviť premenlivú veľkosť medzery, ktorá sa bude náhodne vyberať medzi nulovou a maximálnou veľkosťou medzery, ktorú užívateľ nastaví. Túto funkcionalitu si môže užívateľ zvoliť pomocou prepínaču `Random Offset`.



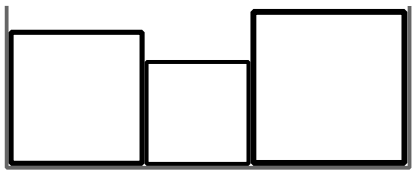
Obr. 3.3: Príklad Random Offset, kedy vzdialenosť medzi jednotlivými objektmi je zvolená náhodne z určitého rozsahu.

Ďalším parametrom je `Random Rotation` (obrázok 3.4), ktorý umožňuje otočenie generovaných objektov v daných osách. Táto rotácia bude náhodné zvolená v rozsahu, ktorý užívateľ zvolí. Pri tomto parametre je možné si zvoliť, okolo ktorých ôs sa bude táto rotácia vykonávať. Rotácia bude vplývať iba na objekt, nie na jeho obal. Tým pádom môžu niektoré objekty po otočení zaberať viac miesta.



Obr. 3.4: Príklad Random Rotation, pri ktorom sa zmení rotácia objektu, čo spôsobí zmenu veľkosti jeho obalu orientovaného podľa ôs scény.

Posledným voliteľným parametrom je `Random Scale` (obrázok 3.5), ktorý umožňuje rozšíreniu meniť veľkosť generovaných objektov náhodne v predom stanovenom rozmedzí. Použitím tohto parametru bude možné vyplniť aj menšie plochy, bez toho aby bolo potrebné vymodelovať nové objekty.



Obr. 3.5: Príklad Random Scale, kedy sa veľkosť vytvorených objektov zvolí náhodne v určitom rozsahu zadanom užívateľom.

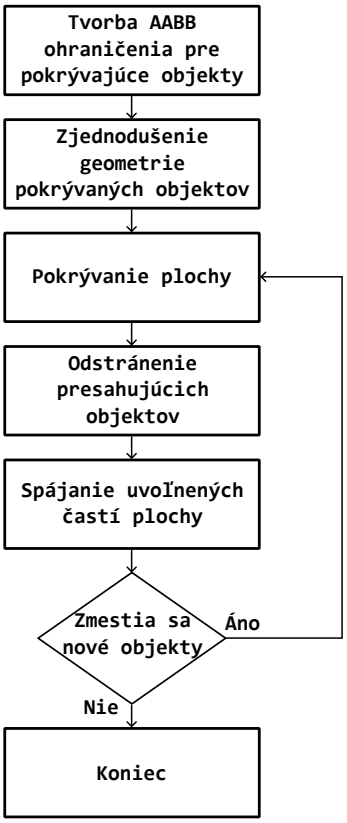
3.4 Generovanie objektov

Funkcionalita tohto rozšírenia (obrázok 3.6) sa skladá z viacerých častí. Ako prvé je potrebné, aby sa všetky objekty a plochy pripravili. Až po dokončení tejto prípravy začne ukladanie týchto objektov do plôch. Po ukončení behu operátoru bude vygenerované množstvo objektov, ktoré budú pokrývať vybraté plochy.

3.4.1 Príprava objektov

Objekty, ktoré budú slúžiť na pokrytie plochy nemajú žiadne obmedzenia ohľadom komplexnosti ich geometrie. To znamená, že objekty môžu byť konvexné, aj konkávne. Z tohto dôvodu by počítanie nimi pokrytej plochy bolo veľmi zložité. Preto sa jednotlivé objekty obaľujú do takzvaných ohraničujúcich objektov (sekcia 2.3). Aplikácia Blender natívne podporuje vytváranie ohraničení rôznych tvarov pre objekty. Z týchto tvarov bolo pre účely tohto rozšírenia zvolené ohraničenie pomocou 3D kvádru (obrázok 3.7).

Z pohľadu pokrývania plochy sa priestor obsadený jedným objektom počíta iba vo dvoch osách 3D priestoru. Tretia os, v tomto prípade výška objektu môže byť zanedbaná. Preto sa pri vypĺňaní plôch bude toto ohraničenie považovať za štvorec, prípadne obdĺžnik. Ohraničenie, ktoré vznikne okolo každého objektu, bude mať svoju rotáciu definovanú podľa ôs priestoru, nie podľa rotácie daného objektu. Tým pádom sa jedná o Axis-Aligned Bounding



Obr. 3.6: Schéma zobrazuje postupnosť operácii, ktoré sa budú vykonávať počas behu rozšírenia.

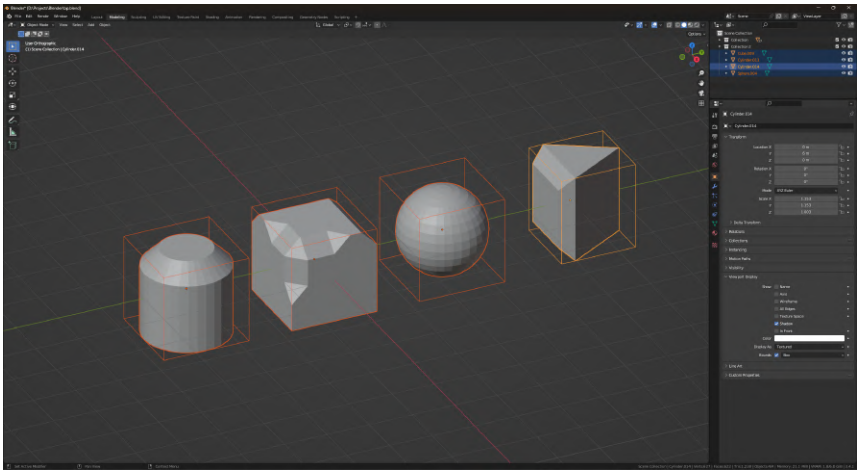
Box (sekcia 2.3.2). Vďaka tomuto ohraničeniu sa výrazne zjednoduší výpočet plochy pokrytej jedným objektom. Ale zároveň sa zníži hustota uloženia objektov na celom polygóne, keďže jednotlivé objekty môžu vytvárať väčšie množstvo nevyužitého priestoru vo vnútri ohraničenia.

Aj keď voliteľné parametre vplývajú na veľkosť a tvar ohraničenia, tak z dôvodu ich určitej náhodnosti, ktorá sa využíva pri aplikovaní týchto parametrov, nie je možné ich zohľadniť už počas prípravy objektov. Z toho dôvodu sa aplikovanie týchto parametrov rieši až počas vypĺňania plôch.

3.4.2 Príprava polygónov kontajnerov

Rovnako, ako pri objektoch určených na pokrývanie, neexistuje žiadne obmedzenie komplexnosti geometrie ani pri plochách objektov, ktoré budú pokrývané. To znamená, že na daných objektoch sa môžu nachádzať polygóny, ktoré nemajú dostatočnú veľkosť na to, aby ich bolo možné pokryť aspoň jedným objektom. Zároveň môžu tieto polygóny byť tvorené rôznym počtom bodov a môžu mať rôzny tvar, čo môže spôsobiť ďalšie obmedzenia pri ich pokrývaní.

Z tohto dôvodu je vhodné upraviť pokrývané objekty predtým, než sa na ich plochy budú vytvárať objekty. Cieľom tejto úpravy bude znížiť množstvo polygónov objektu pomocou ich zjednotenia. Polygóny, ktoré budú spojené, by mali mať všetky body v jednej rovine,



Obr. 3.7: Využitie 3D Bounding Boxu v aplikácii Blender

aby táto úprava nespôsobila zmenu geometrie objektu. Vďaka tomu vzniknú na objekte väčšie plochy, ktoré bude možné hustejšie pokryť objektmi.

3.4.3 Vypĺňanie plochy

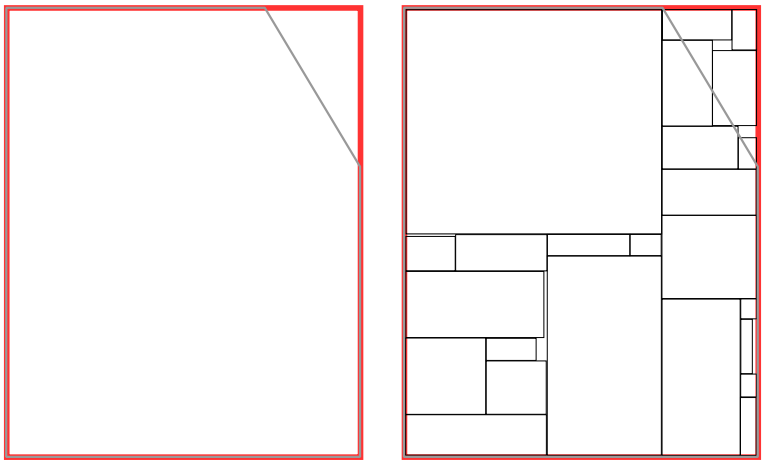
Jednotlivé polygóny pokrývaných objektov sa môžu skladať z rôzneho počtu bodov a môžu mať rôzny tvar. Tento tvar nemusí byť pravidelný, čím sa zvyšuje náročnosť pokrývania tohto povrchu. Z toho dôvodu sú jednotlivé pokrývané plochy obaľované do ohraničujúcich objektov. Jedná sa o ohraničenie pomocou 2D Axis-Aligned Bounding Box (sekcia 2.3.2). Vďaka tomu sa z tejto optimalizačnej úlohy stáva problém ukladania pravidelných objektov do pravidelných kontajnerov.

Rozšírenie prechádza cez všetky polygóny vypĺňaných objektov a pokrýva ich v rade za sebou. Pri každom polygóne prichádza najprv k jeho ohraničeniu. Po vytvorení jeho ohraničenia dochádza k pokrývaniu plochy ohraničujúceho tvaru objektmi (obrázok 3.8). Objekty, ktoré sú využívané na pokrývanie plôch sú vyberané náhodne pomocou pseudonáhodného generátoru čísel (sekcia 2.2) zo zoznamu všetkých objektov vybratých užívateľom.

Po vytvorení každého objektu sa naň aplikujú všetky aktivované voliteľné parametre tohto rozšírenia. Tieto parametre umožňujú modifikovať veľkosť, otočenie objektov a vzdialenosť medzi jednotlivými vygenerovanými objektmi. Počet inštancií každého jedného takéhoto objektu je neobmedzený. Každý polygón pokrývaného objektu sa vypĺňa až do bodu, kedy do neho nie je možné zmestiť ďalší objekt. Objekty, ktoré sa už nedokážu zmestiť do aktuálne pokrývaného polygónu sú vyradené zo zoznamu výberu. Keď tento zoznam zostane prázdny, dochádza k ukončeniu prvotného pokrývania plochy.

Po ukončení tohto pokrývania sa začne vykonávať kontrola všetkých objektov vytvorených na pokrytie tohto polygónu. Jej cieľom je zistiť, či sa všetky vytvorené objekty, pokrývajúce ohraničujúci tvar, nachádzajú aj vo vnútri polygónu. V prípade, že sa nájdu objekty, ktoré prekračujú hranice tohto polygónu, sú vymazané (obrázok 3.9). Avšak vymazávanie týchto objektov môže vytvoriť oblasti polygónu, ktoré už nebudú pokryté žiadnymi objektmi. Jednotlivé tieto plochy budú najprv skontrolované, či nie je možné ich spojiť a vytvoriť tým menší počet väčších oblastí, potom budú ďalej spracované.

S každou z týchto oblastí sa pracuje ako s polygónom. To znamená, že pre každú oblasť sa bude znovu vykonávať jej pokrývanie, ktoré bude fungovať rovnakým spôsobom, ako



Obr. 3.8: Práve pokrývaný polygón sa najprv obalí do pravidelného ohraničujúceho tvaru. Potom je objektmi najprv pokrytý celý ohraničujúci tvar, nie len polygón.

pri pôvodnom polygóne (obrázok 3.10). Zanorovanie týmto spôsobom bude prebiehať až do bodu, kedy vzniknú oblasti s príliš malou veľkosťou na to, aby bolo možné do nich vložiť aspoň jeden objekt.

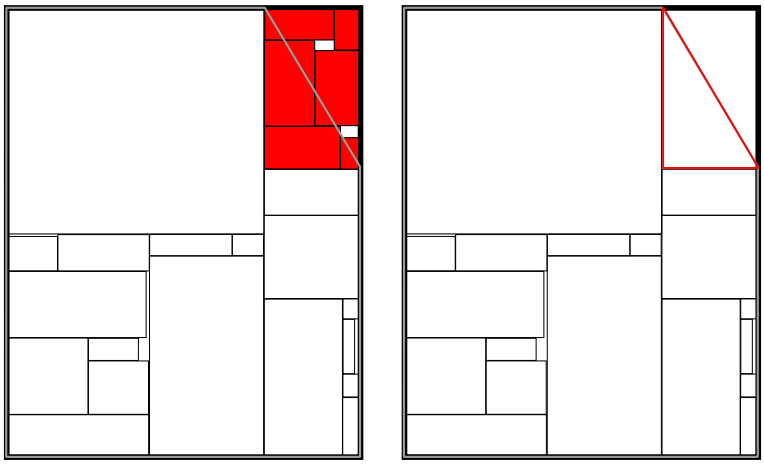
Týmto spôsobom sa postupne vyplní každý jeden polygón všetkých pokrývaných objektov. Keďže každé jedno pokrývanie plochy končí vyprázdnením zoznamu pokrývajúcich objektov, tento zoznam musí byť zálohovaný a vždy pri prechode medzi polygónmi, prípadne jeho oblasťami sa musí znovu obnoviť. Po ukončení pokrývania všetkých polygónov sa beh rozšírenia skončí.

3.4.4 Výstup operátoru

Výstupom tohto operátoru je množina objektov vytvorených za účelom pokrytia užívateľom zvolených plôch. Z toho dôvodu, že neexistuje žiadne obmedzenie na počet pokrývaných objektov, je vhodné spraviť určitú hierarchiu, do ktorej sa tieto vygenerované objekty budú ukladať. V Blenderi existujú kolekcie, ktoré umožňujú zoskupovanie rôznych objektov za účelom zprehľadnenia scény a jej hierarchie. Tieto kolekcie je možné využiť na to, aby objekty pokrývajúce plochy jedného objektu boli v hierarchí zoskupené do jedného priečinku pod jedným názvom. Vďaka tomu bude výstup rozšírenia výrazne prehľadnejší v prípade, že sa užívateľ bude snažiť dostať k určitej časti objektov.

Avšak stále by mohlo prísť k zahlteniu hierarchie scény veľkým množstvom kolekcií, ktoré môžu vzniknúť ako výstup tohto rozšírenia. Aby sa tomuto zahlteniu predišlo, tak by sa všetky tieto kolekcie vytvorené pre jednotlivé pokrývané objekty vkladali do jednej hlavnej kolekcie (obrázok 3.11). Táto kolekcia bude predstavovať celkový výstup tohto operátoru.

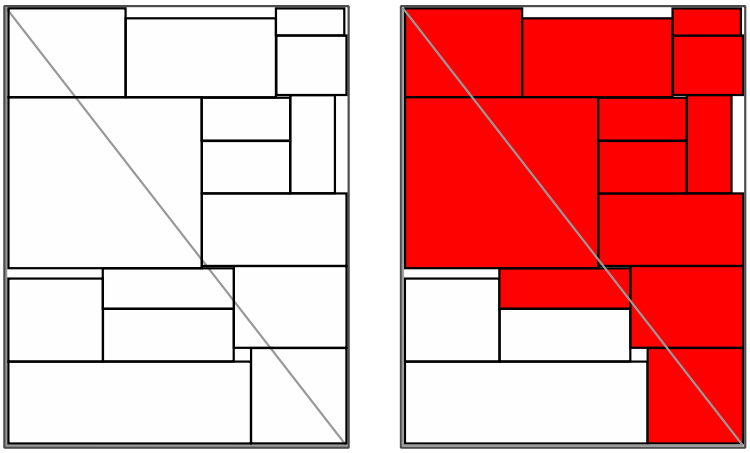
Pri nastavovaní parametrov rozšírenia môže byť operátor užívateľom spustený viac krát. Pri každom tomto spustení sa vygeneruje nová kolekcia, ktorá bude obsahovať jeho výstup. Týmto spôsobom by mohlo prísť k zahlteniu scény veľkým množstvom objektov vzniknutých pri viacerých behoch operátoru. Z toho dôvodu je vhodné regulovať množstvo týchto



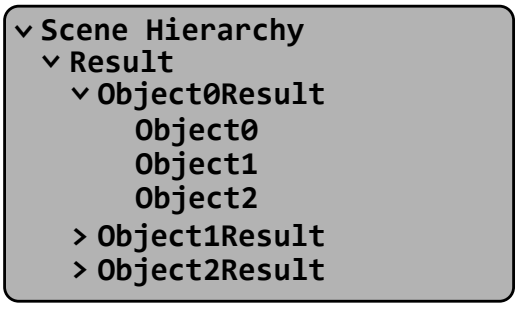
Obr. 3.9: Po dokončení prvotného pokrývania objektov nastáva ich kontrola. Zo všetkých vytvorených objektov sa vyberú tie, ktoré presahujú hranice pôvodného polygónu a vymažú sa.

vzniknutých kolekcií. Udržiavať sa bude vždy iba posledná vzniknutá kolekcia obsahujúca výstup operátoru. V prípade, že užívateľ znovu spustí operátor, tak sa všetky objekty a kolekcie v tejto kolekcií vymažú a vytvorí sa nová kolekcia obsahujúca výstup aktuálneho behu operátoru.

Zároveň je v určitých prípadoch potrebné, aby si užívateľ dokázal uložiť výstupnú kolekciu, ktorá by sa ďalším spustením operátoru neprepísala. Z toho dôvodu bude vymazávanie predchádzajúcich výstupov obmedzené na pôvodný názov kolekcie, ktorý je priradený pri jej vytvorení. V prípade, že si užívateľ bude chcieť permanentne uložiť výstup operátoru, tak bude k tomu stačiť premenovanie výstupnej kolekcie.



Obr. 3.10: Nevyplnená oblasť, ktorá vznikla vymazávaním presahujúcich objektov, sa považuje za polygón a prebieha jej obaľovanie do ohraničujúceho tvaru a následné pokrytie objektmi. Ďalej bude prebiehať znovu kontrola vytvorených objektov.



Obr. 3.11: Ukážka príkladu hierarchie scény obsahujúca výsledok behu rozšírenia.

Kapitola 4

Implementácia

Na tvorbu rozšírení ponúka aplikácie Blender vstavaný textový editor, ktorý umožňuje programovanie a spúšťanie rozšírení priamo v aplikácii. Tento editor je priamo prepojený s interpretom jazyka Python, pomocou ktorého je možné implementovať ľubovolné rozšírenia. Na zabezpečenie správnej komunikácie medzi jazykom Python a aplikáciou Blender existuje aplikačné rozhranie Blender API, ktoré poskytuje prístup k dátovým objektom a operáciám Blenderu. Pomocou tohto rozhrania je možné implementovať funkcionalitu rozšírenia a zároveň aj jeho užívateľské rozhranie.

Na implementáciu tohto rozšírenia bolo použité predpripravené prostredie určené na implementáciu a testovanie rozšírení, ktoré sa nachádza natívne v aplikácii Blender. Toto prostredie umožňuje priame testovanie rozšírení pomocou ich jednorázového okamžitého spustenia cez vstavaný prekladač. Okrem toho je možné rozšírenia spúšťať ich permanentným pridaním medzi inštalované rozšírenia cez nastavenia aplikácie.

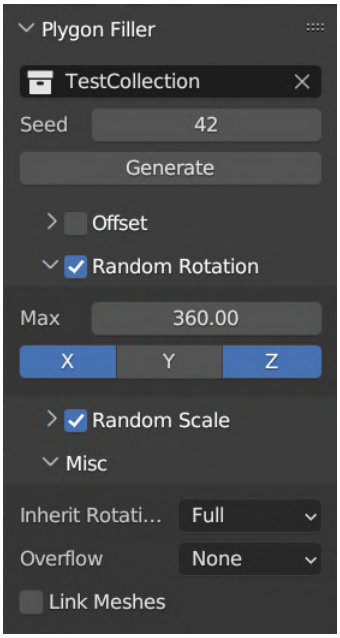
V prílohách je možné nájsť návod (príloha A), ktorý popisuje inštláciu rozšírenia a príklady scén (príloha B), ktoré boli vytvorené pomocou tohto rozšírenia.

4.1 Užívateľské prostredie

Pri implementácii užívateľského prostredia boli vykonané určité zmeny v jeho rozložení (obrázok 4.1). Časť týchto zmien bola vykonaná kvôli spôsobu, akým funguje hierarchia prvkov užívateľského prostredia v rozhraní Blender API. Ostatné zmeny boli vykonané za účelom zjednodušenia a urýchlenia práce s rozšírením, prípadne aby pokryli novo pridanú funkcionalitu rozhrania, ktorá nebola súčasťou pôvodného návrhu a jej potreba sa preukázala až pri implementácii.

Hierarchia prvkov užívateľského prostredia v Blender API zapríčinila, že bolo potrebné zmeniť poradie jednotlivých panelov rozhrania. Podpanely nie je možné v Blenderi vložiť pred ostatné prvky, ako napríklad parametre alebo operátory, ktoré sa majú nachádzať v hlavnom panely. Z toho dôvodu bolo základné ovládanie rozšírenia presunuté na začiatok užívateľského rozhrania. Pod základným ovládaním sa nachádzajú podpanely obsahujúce ovládanie jednotlivých užívateľských parametrov. Všetky tieto podpanely je možné v prípade ich nepotrebnosti skryť, aby nezaberali miesto na obrazovke.

Za cieľom uľahčenia práce s týmto rozšírením bol prerobený spôsob vyberania pokrývajúcich objektov. Spočiatku bol tento výber navrhnutý ako zoznam, do ktorého by bolo možné pridávať a odstraňovať jednotlivé objekty. Tento spôsob bol vymenený za výber pokrývajúcich objektov pomocou jednej kolekcie, ktorá bude všetky tieto objekty obsahovať.



Obr. 4.1: Výsledný dizajn užívateľského prostredia zobrazujúci zmeny od pôvodného návrhu a možnosť skrývať podpanely.

Táto zmena umožňuje vyberanie väčšieho množstva objektov cez jednu kolekciu namiesto toho, aby bol užívateľ nútený pridávať objekty do zoznamu po jednom.

Poslednou vykonanou zmenou bolo pridanie nového podpanelu, ktorý obsahuje novo pridané užívateľské parametre. Tento podpanel nie je možné deaktivovať rovnakým spôsobom ako panely ostatných parametrov, pretože jednotlivé parametre, ktoré sa v ňom nachádzajú pracujú nezáväzne na sebe.

4.2 Príprava objektov

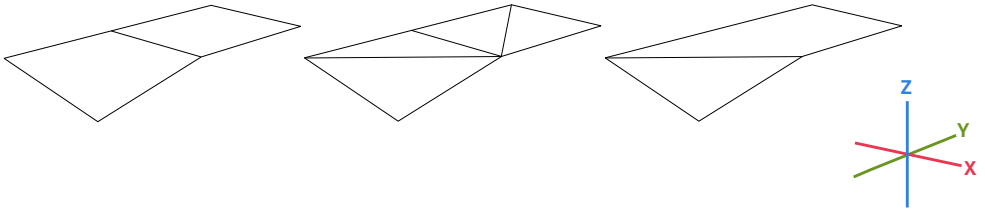
Pred generovaním pokrytia objektov je potrebné pripraviť všetky vybraté pokrývané a pokrývajúce objekty. Cieľom tejto prípravy je zabránenie neočakávaného správania spôsobeného neočakávanými hodnotami, ktoré môžu byť nastavené v niektorých užívateľom zvolených objektoch. V prípade, že nastávajú pri príprave zmeny geometrie, tak sa tieto zmeny vykonávajú nad kópiami pôvodných objektov, aby sa zabránilo strate pôvodných, užívateľom vytvorených objektov.

4.2.1 Pokrývajúce objekty

Pri príprave pokrývajúcich objektoch nedochádza k zmene ich geometrie, ale iba určitých parametrov. Týmto objektom je kvôli zjednodušeniu výpočtov pri pokrývaní nastavené ohraničenie v tvare 3D kvádra. Ďalej sa počas prípravy ukladajú odkazy na jednotlivé tieto objekty do poľa, s ktorým sa potom manipuluje pri generovaní ich kópii počas pokrývania plôch. Toto pomocné pole je potrebné z dôvodu zmeny dostupných objektov na pokrývanie častí jednotlivých plôch, ktoré nastávajú počas behu rozšírenia.

4.2.2 Pokrývané objekty

Príprava pokrývaných objektov zahrňuje viacero krokov, ktoré by permanentne zmenili geometriu užívateľom vytvorených objektov. Z toho dôvodu sa vytvorí ich kópia, ktorá je pri konci behu operátoru `Generate` odstránená. Oproti návrhu pribudlo použitie modifikátoru `Triangulate`. Cieľom tohto modifikátoru je rozdeliť všetky polygóny objektu na trojuholníky. Tento modifikátor je použitý v spojení s modifikátorom `Decimate` v nastavení `Planar`, ktorý spojí všetky polygóny, ktoré zvierajú uhol menší ako nastavený prah. Tento prah je nastavený na veľmi malú hodnotu. Týmto spôsobom je možné nahradiť všetky polygóny tvorené bodmi, ktoré sa nenachádzajú v jednej rovine, skupinou trojuholníkov a polygónov, ktorých body tvoria jednu rovinu (obrázok 4.2).



Obr. 4.2: Postupnosť procesu prípravy pokrývaného objektu. Vstupné polygóny nemusia mať všetky body v jednej rovine (ľavý objekt), preto je na pokrývané objekty aplikovaný modifikátor Triangulate (stredný objekt) a potom ešte aj modifikátor Decimate (pravý objekt). Vďaka tomu bude mať každý polygón všetky body v jednej rovine.

Ďalej je potrebné obnoviť pôvodné hodnoty parametrov pozície, otočenia a veľkosti týchto objektov. Posledná zmena, ktorá je počas prípravy vykonaná na pokrývaných objektoch je zrušenie prepojenia hrán medzi polygónmi, aby jednotlivé polygóny nezdieľali tie isté body. Toto rozdelenie je vykonané z dôvodu ďalšej manipulácie vykonávanej nad jednotlivými polygónmi počas ich pokrývania.

4.3 Generovanie objektov

Výsledkom každého behu tohto rozšírenia je hierarchia kolekcií, ktorá obsahuje jednotlivé vygenerované objekty. Aby sa predišlo nadmerného zahlteniu scény, tak algoritmus najprv prechádza výsledky predchádzajúceho behu a odstraňuje ich. Hlavnú kolekciu s výsledkom predchádzajúceho behu rozšírenie hľadá podľa jej pevne daného názvu. V prípade, že by si chcel užívateľ ponechať jeden z výsledkov behov, stačí premenovať jeho hlavnú kolekciu. Po jej premenovaní už nebude táto kolekcia nájdená počas nového behu a preto nebude odstránená.

Ďalšie úkony, ktoré musí algoritmus na začiatku behu vykonať je inicializovanie generátoru náhodných čísel, príprava objektov (sekcia 4.2) a vytvorenie kolekcie na ukladanie vytvorených objektov. Generátor náhodných čísel je inicializovaný hodnotou, ktorý si užívateľ zvolí pomocou parametru `seed`. Názov vytvorenej kolekcie je pevne zadaný v kóde rozšírenia.

Počas jedného behu je možné pokrývať viacero objektov. Algoritmus postupne prechádza cez všetky užívateľom zvolené objekty a pokúša sa ich pokryť čo najhustejšie, vzhľadom na užívateľom nastavené parametre. Pre každý jeden objekt prechádza algoritmus cez všetky jeho plochy (algoritmus 1). Pre každú túto plochu vytvorí orientované ohraničenie (sekcia 2.3.3), ktoré je potom pomocou binárneho stromu pokryté objektmi. Avšak, aby sa pokryla

iba plocha tvaru polygónu a nie jeho ohraničenia, tak sú jednotlivé objekty kontrolované, či neprekračujú hranice plochy. V prípade, že tieto hranice prekračujú, tak spôsob, ktorým sa s týmito objektmi algoritmus vysporiada, je založený na nastavení užívateľského parametru `overflow`.

Algorithm 1 Operator Generate

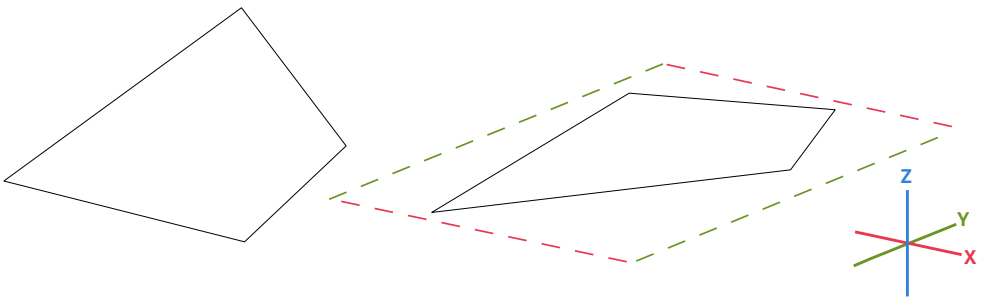
Input: Collection of objects, Objects, User parameters
Output: Polygons covered with objects from collection

```
delete_previous_result()
init_number_generator()
prepare_objects()
prepare_containers()
for all container_objects do
  for all polygons do
    rotate_polygon()
    create_bounds()
    fill_polygon()
  end for
end for
```

4.3.1 Rotácia polygónu

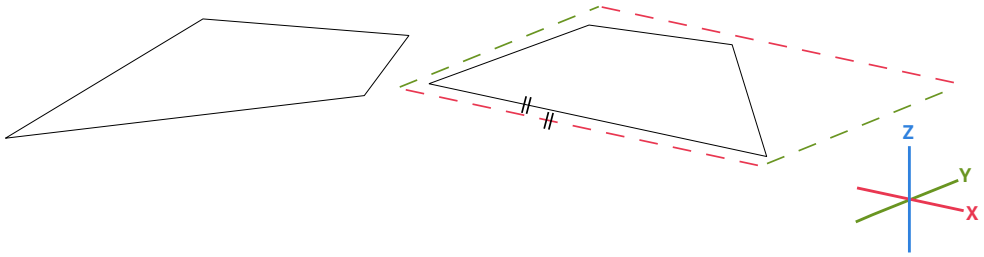
Jednotlivé polygóny nachádzajúce sa na pokrývaných objektov môžu mať ľubovoľnú rotáciu a tvar. Tieto parametre plochy môžu negatívne vplývať na zložitosť výpočtov pozícií pre vygenerované objekty. Za účelom zjednodušenia týchto výpočtov si upraví algoritmus rotáciu jednotlivých plôch. Vďaka tejto úprave je možné pre danú plochu vytvoriť orientované ohraničenie a zároveň zostane plocha v takom stave, že si ponecháva tvar a rozmer pôvodnej plochy, ale je rovnobežná s rovinou XY. Táto úprava umožňuje zanedbať pozíciu objektov na osi Z pri výpočtoch ich pozícií v rámci pokrytia.

Tieto rotácie je možné na jednotlivé plochy aplikovať vďaka oddeleniu jednotlivých plôch objektov, ktoré bolo vykonané počas prípravy pokrývaných objektov. Prvá rotácia, ktorá sa na objekt použije je jej zarovnanie do roviny XY (obrázok 4.3). Táto rotácia je vypočítaná pomocou uhlu, ktorý zviera normálový vektor plochy a normálový vektor roviny XY.



Obr. 4.3: Prvá rotácia ktorá je na pôvodný polygón (ľavý objekt) aplikovaná je vypočítaná pomocou normálového vektoru polygónu a roviny XY. Výsledkom je objekt zarovnaný do roviny XY (pravý objekt).

Účel ďalšej rotácie použitej na plochu je vytvorenie orientovaného ohraničenia. Veľkosť tejto rotácie sa počíta z uhlu, ktorý zviera najdlhšia hrana plochy s osou X. Výsledkom tejto rotácie je polygón, ktorého najdlhšia hrana je rovnobežná s osou X (obrázok 4.4). V určitých prípadoch táto rotácie spôsobí zmenu orientácie plochy, príčinou čoho stratí plocha jej zarovnanie s rovinou XY. Namiesto toho bude plocha kolmá na túto rovina. Z toho dôvodu sa na plochu znovu aplikuje rotácia vypočítaná z uhlu zvieraného normálovým vektorom plochy a normálovým vektorom roviny XY.



Obr. 4.4: Druha aplikovaná rotácia otáča objekt vytvorený pomocou prvej rotácie (ľavý objekt) podľa osi Z. Výsledkom tejto rotácie je objekt, ktorého najdlhšia hrana je rovnobežná s osou X (pravý objekt). Z tohto polygónu je potom možné vytvoriť orientované ohraničenie.

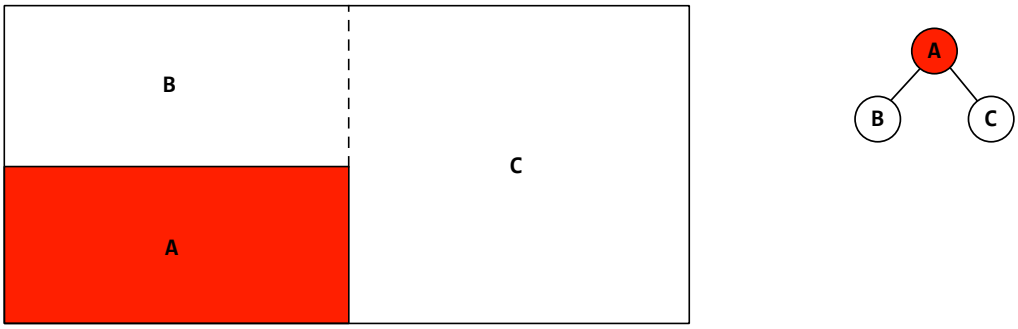
Po aplikovaní týchto troch rotácii sa vypočíta osovo-orientované ohraničenie tejto plochy, ktoré by sa v prípade otočenia polygónu do pôvodnej rotácie správalo ako orientované ohraničenie objektu. Jednotlivé hrany tohto ohraničenia sú definované maximálnou a minimálnou hodnotou na osách X a Y, ktoré nadobúdajú jednotlivé body tvoriace plochu.

Pred začatím pokrývania sa vykonáva ešte jedna kontrola zameraná na pozíciu najdlhšej hrany plochy v rámci jej ohraničenia. V prípade, že sa táto hrana nachádza na hornej hranici ohraničenia, tak je celá plocha ešte otočená podľa osi Z o 180 stupňov. Týmto sa predíde nekonzistencii, ktorá by mohla vznikať pri plochách objektov, ktoré majú rovnaký tvar a veľkosť, ale odlišnú rotáciu v rámci 3D priestoru scény.

4.3.2 Pokrývanie pomocou binárneho stromu

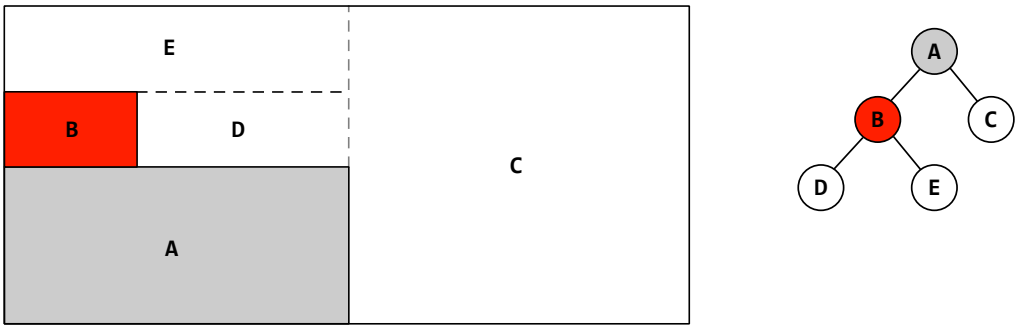
Pokrývanie plôch objektov je implementované rekurzívne pomocou binárneho stromu. Na začiatku pokrývania má algoritmus dostupnú celú plochu ohraničenia, ktoré má tvar štvorca, prípadne obdĺžniku. Táto plocha je pre potreby pokrývania definované jej počiatkom, to znamená pozíciou jej ľavého dolného bodu, a jej šírkou a výškou. Pri ďalších spusteniach tejto funkcie v rekurzií sa počíta už len s podmnožinou tejto plochy, ktorá je vypočítaná z veľkosti plochy vyššej úrovne a veľkosti vygenerovaného objektu.

Pri každom behu tejto funkcie sa vygeneruje jeden objekt. Tento objekt je náhodné pomocou generátoru náhodných čísel zvolený z kópie listu objektov vybratých užívateľom na pokrývanie plôch. Po vytvorení objektu sa obnovia do pôvodnej hodnoty jeho hlavné parametre - pozícia, rotácia a veľkosť. Ďalej sú v prípade ich aktivovania na objekt aplikované parametre `Random Scale` a `Random rotation`. Potom nasleduje ďalšie obnovenie hlavných parametrov za účelom aktualizovať rozmery ohraničenia vytvoreného objektu, ktoré sú potom použité na výpočet veľkosti častí plôch, ktoré budú pokrývané v ďalšom zanorení tejto funkcie. Počas týchto výpočtov sa berie už aj ohľadom na parameter `Offset`, v prípade, že bol užívateľom aktivovaný.



Obr. 4.5: Vypĺňanie plochy pomocou binárneho stromu pre časti plochy, ktorých počiatočný bod leží na spodnej hranici ohraničenia polygónu. Po vytvorení objektu je zvyšná plocha rozdelená na dve časti, podľa šírky objektu.

Po vytvorení objektu sa funkcia zanorí pomocou rekurzie, kde vytvorí ďalšie dva behy tej istej funkcie. Plocha, ktorá bude predaná do týchto zanorených volaní môže byť vypočítaná dvoma spôsobmi. Zvolený spôsob závisí od toho, či aktuálny beh funkcie bol zavolaný pre časť plochu, ktorej počiatočný bod leží na spodnej hranici ohraničenia. V prípade, že počiatočný bod na nej leží, tak je plocha rozdelená podľa šírky vygenerovaného objektu (obrázok 4.5). V opačnom prípade, kedy sa počiatočný bod objektu nenachádza na spodnej hranici ohraničenia, tak je plocha rozdelená podľa výšky vygenerovaného objektu (obrázok 4.6). Cieľom tejto zmeny je zabrániť nižšej rôznorodosti vygenerovaných objektov, ktorá by mohla vzniknúť v prípade, že bude na pokrytie plochy vybratý jeden z menších užívateľom zvolených objektov.



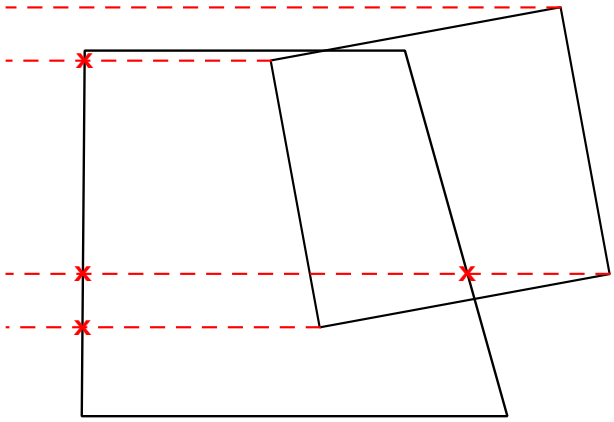
Obr. 4.6: Vypĺňanie plochy pomocou binárneho stromu pre časti plochy, ktorých počiatočný bod neleží na spodnej hranici ohraničenia polygónu. Po vytvorení objektu je zvyšná plocha rozdelená na dve časti, podľa výšky objektu.

4.3.3 Kontrola presahovania objektov

Po zavolaní zanorenia prebieha ešte pre každý vygenerovaný objekt kontrola, ktorej cieľom je odhaliť, či daný objekt nepresahuje hranice pôvodnej plochy. Princíp, na ktorom je táto kontrola založená spočíva vo vytvorení 4 pomyselných bodov. Tieto body sú definované ohraničením objektu, avšak ich pozícia v scéne je mierne upravená. Táto úprava spôsobuje, že plocha definovaná týmito štyrmi bodmi je v obidvoch smeroch o 1 percento menšia a podľa osi Z sa nachádza vo výške aktuálne pokrývanej plochy. Cieľom zmenšenia tejto

plochy je zabránenie falošných hlásení presahujúcich bodov, ktoré by mohli byť spôsobené nepresnosťou floatových hodnôt definujúcich pozíciu jednotlivých bodov.

Ďalej sa na tieto body použije algoritmus `ray casting`, ktorý pre každý bod počíta počet pretnutí priamky, ktorá začína v danom bode a pokračuje rovnobežne s osou X, a jednotlivých hrán plochy (obrázok 4.7). V prípade, že je tento počet párny alebo rovný nule, tak sa daný bod považuje sa presahujúci a je uložený do zoznamu presahujúcich bodov daného vygenerovaného objektu.



Obr. 4.7: Kontrola presahovanie hraničných bodov vygenerovaného objektu. Z každého hraničného bodu objektu sa vytvárajú priamky, pri ktorých sa počítajú ich pretnutia s hranami polygónu. Pre body mimo polygónu je tento počet párny, prípadne rovný nule.

Spôsob, akým sa bude rozšírenie vysporiadávať s presahujúcimi bodmi závisí od viacerých parametrov. Objekty, pri ktorých presahujú všetky štyri body, sú automaticky vymazané zo scény. Presahovanie ostatných objektov, pri ktorých presahuje len časť ich bodov, je závislá od nastavenia užívateľského parametru `Overflow`, s ktorým bolo rozšírenie spustené.

4.3.4 Výsledná pozícia objektov

Po prefiltrovaní vygenerovaných objektov pomocou kontroly presahovania pokrývanej plochy je potrebné týmto objektom nastaviť takú pozíciu, aby sa nachádzali vo vnútri pôvodnej, užívateľom vytvorenej plochy a nie vo vnútri jej modifikovanej verzie. Počas aplikovania rotácii na pôvodnú plochu sa ukladali jednotlivé aplikované rotácia do poľa. Vďaka tomuto poľu hodnôt je možné získať výslednú pozíciu objektu aplikovaním všetkých rotácií na pozíciu objektu v opačnom poradí, než boli aplikované pri modifikovaní plochy.

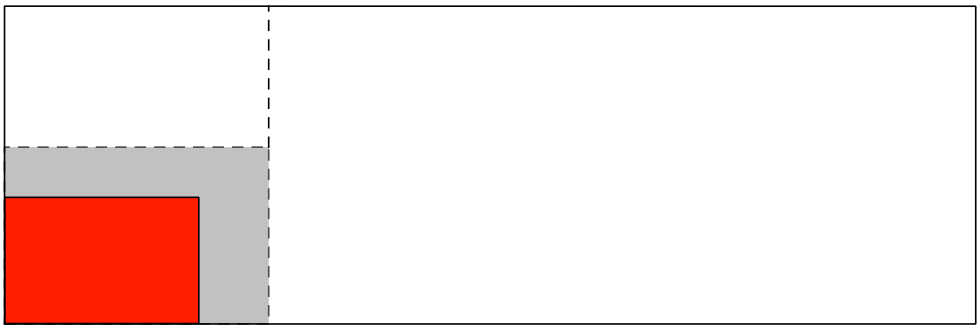
4.4 Užívateľské parametre

Výsledok behu rozšírenia je možné upraviť pomocou parametrov. Tieto parametre sa postupne počas behu aplikujú na jednotlivé vytvorené objekty a menia ich vlastnosti. Ďalej môžu tieto parametre určovať správanie algoritmu v prípade, že vytvorené objekty nespĺňajú ideálne podmienky, ako napríklad, že presahujú za hranice plochy.

4.4.1 Offset

Parameter `Offset` umožňuje užívateľovi vytvoriť medzery medzi jednotlivými objektmi alebo umožňuje prekrytie vytvorených objektov. Veľkosť týchto medzier je nastavená uží-

vateľom ešte pred spustením rozšírenia. V prípade, že užívateľ nastaví zápornú hodnotu pre parameter `Offset` v určitej osi, tak sa jednotlivé objekty budú prekrývať. Je možné nastaviť rôznu veľkosť offsetu pre každú osu v 3D priestore. Osi, podľa ktorých sa tento offset aplikuje patria priestoru definovanému pokrývaným polygónom. Vďaka tomu je možné využiť osi X a Y na tvorbu medzier (obrázok 4.8), prípadne prekrytia medzi objektmi a pomocou offsetu na ose Z nastaviť výšku objektu vzhľadom na plochu.



Obr. 4.8: Príklad využitia parametru Offset počas behu rozšírenia. Objekt je vytvorený na počiatočnom bode časti plochy. Zvyšok plochy je ďalej rozdelený podľa výšky/šírky objektu s pripočítaným offsetom.

Tento parameter je ďalej možné modifikovať pomocou nastavenia `Random Offset`, ktoré náhodne vytvára priestory medzi objektmi podľa užívateľom stanovených parametrov. Hodnota, ktorá sa pre každú osu náhodne zvolí je v rozsahu 0 až hodnota, ktorú nastaví užívateľ. Táto hodnota môže byť kladná aj záporná. Náhodne vybraná hodnota offsetu je zvolená pomocou toho istého pseudonáhodného generátoru čísel, ktorý sa používa aj na generovanie objektov.

Pri použití nastavenia `Random Offset` existuje určité obmedzenie. Cieľom tohto obmedzenia je zabrániť vzniku nevyplnených častí plochy. V prípade, že sa objekt nezmestí do práve pokrývanej časti plochy následkom príliš veľkej náhodne zvolenej veľkosti offsetu, tak sa tento offset v osách X a Y nastaví na maximálnu hodnotu, s ktorou sa ešte dokáže objekt zmestiť do pokrývanej časti plochy.

4.4.2 Random Rotation

Tento parameter umožňuje užívateľovi aplikovať dodatočnú rotáciu na vytvorené objekty. Veľkosť tejto rotácie je zvolená náhodne pomocou psuedonáhodného generátoru čísel. Rozsah rotácie je 0 až užívateľom zvolená kladná hodnota. Maximálna hodnota, ktorú môže užívateľ zadať je 360 stupňov. Ďalej je možné aplikovať túto rotáciu iba na zvolené osi. Pre všetky zvolené osi je rozsah rotácii rovnaký, avšak generátor čísel vytvorí pre každú osu hodnotu zvlášť.

4.4.3 Random Scale

Pomocou parametru `Random Scale` je možné meniť veľkosť vygenerovaných objektov. Rozsah tejto veľkosti si môže nastaviť užívateľ pomocou hodnoty pre minimálnu a maximálnu možnú veľkosť. Z tohto rozsahu sú potom pomocou pseudonáhodného generátaoru čísel vyberané náhodne hodnoty, ktoré sú aplikované na jednotlivé objekty vo všetkých troch osách. Vstupné hodnoty užívateľa sú obmedzené na veľkosť 0.01 až 100 násobok pôvodnej veľkosti

objektov. Zároveň je možné nastaviť obom vstupným parametrom rovnakú hodnotu, čím bude veľkosť všetkých vygenerovaných objektov zmenená na danú hodnotu.

Rovnako ako pri parametri `Random Offset` existuje implementačné obmedzenie na voľbu náhodnej veľkosti objektov. V prípade, že sa vytvorí objekt, ktorý by sa s pôvodnou veľkosťou zmestil do pokrývanej plochy, ale príčinou zvolenia väčšej veľkosti by už presahoval, tak sa obmedzí pre danú časť plochy maximálna možná veľkosť, ktorú môže objekt pomocou generátoru čísel nadobudnúť na veľkosť 1,1. Týmto spôsobom sa zabráni vymazaniu daného objektu zo zoznamu dostupných objektov na pokrývanie, čo by mohlo v krajnom prípade spôsobiť nepokrytie danej časti plochy.

4.4.4 Inherit Rotation

Pri vytváraní pokrytia sa za účelom zjednodušenia výpočtov aplikujú určité rotácie na jednotlivé pokrývané plochy. Tieto rotácie zaručujú, že plocha po ich aplikovaní bude ležať v rovine XY a jej najdlhšia hrana bude rovnobežná s osou X. Toto spôsobuje, že po vrátení vytvorených objektov na pozíciu v rámci pôvodnej plochy, majú tieto objekty odlišnú rotáciu s ohľadom na plochu než mali pri svojom generovaní.

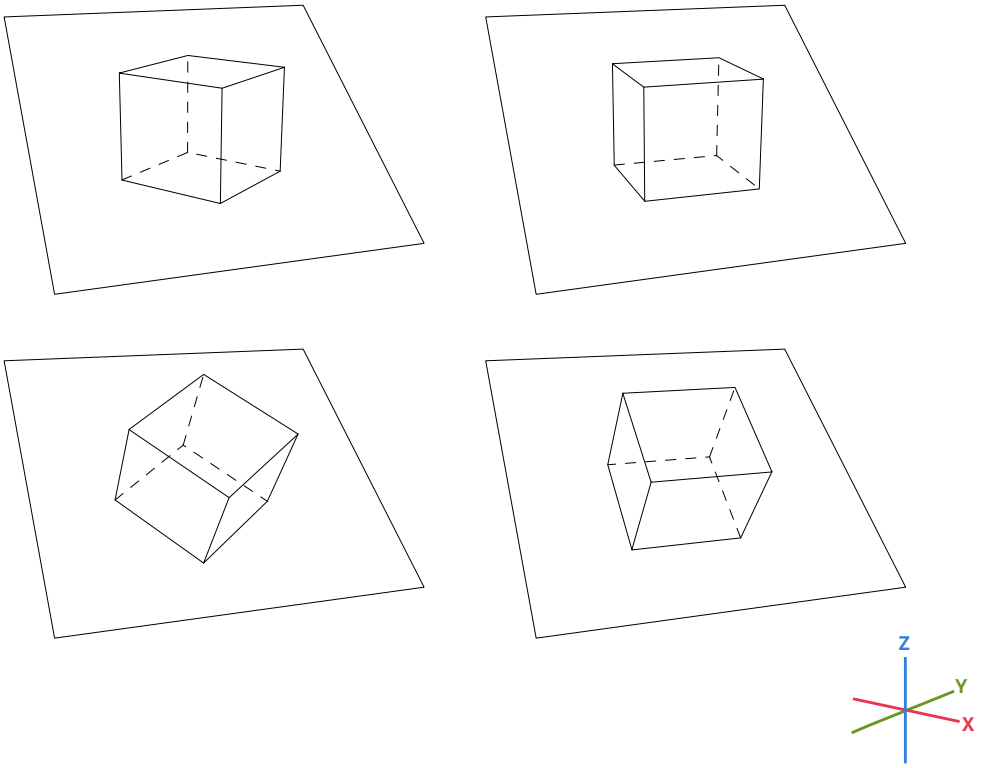
Cieľom tohto parametru je umožniť užívateľovi aplikovať rotácie vytvorené pri otáčaní plochy na jednotlivé objekty. Tento parameter má štyri nastavenia - `None`, `Simple`, `Complex` a `Full` (obrázok 4.9). V nastavení `None` nebude na objekt aplikovaná žiadna z týchto rotácii. `Simple` aplikuje na objekt iba rotáciu podľa osi Z, ktorá vznikla pri otáčaní plochy za účelom vytvorenia rovnobežnosti jej najdlhšej hrany s osou X. Cieľom tohto nastavenia je otočiť objekty tak, aby spodná hrana ich ohraničenia bola rovnobežná s najdlhšou hranou polygónu. Nastavenie `Complex` aplikuje rotácie podľa ôs X a Y. Tieto rotácie boli vytvorené pri otáčaní plochy podľa roviny XY a zaručia že rotácia týchto objektov bude v týchto osách nulová vzhľadom k ploche, ktorú pokrývajú. Posledné nastavenie `Full` je kombináciou dvoch predchádzajúcich nastavení.

4.4.5 Overflow

Tento parameter určuje spôsob, akým sa rozšírenie vysporiada s objektmi, ktoré presahujú hranice polygónu menej ako štyrmi hraničnými bodmi. K dispozícii sú tri nastavenia tohto parametru - `None`, `Sliced` a `Full`.

Nastavenie `None` neumožňuje ponechanie objektov, ktoré presahujú hranice polygónu. Takéto objekty sú vymazané a algoritmus sa pokúsi o ich náhradu inými menšími objektmi. Tento proces sa vykonáva počas generovania jednotlivých objektov na pokrývanie plochy. Objekt, ktorý presahuje hranice polygónu je odstránený zo zoznamu dostupných objektov pre danú časť plochy a aktuálne pokrývaná plocha je zmenšená o 5 percent. Smer, v ktorom sa pokrývaná časť plochy zmenší závisí od počtu presahujúcich bodov a ich pozícii vzhľadom na stred vygenerovaného objektu. V prípade, že presahuje objekt jedným alebo tromi hraničnými bodmi sa zmenší šírka aj výška pokrývanej plochy. Pri presahovaní objektu dvoma bodmi sa zmenší jej veľkosť iba v jednom smere, ktorý je definovaný pozíciou presahujúcich bodov objektu.

Nastavenie `Sliced` nevymazáva objekty, ktoré presahujú hranice polygóny, ale mení ich geometriu za účelom odstránenia tohto presahovania. Táto úprava je dosiahnutá pomocou modifikátoru `Boolean` v nastavení `Difference`. V prípade, že je užívateľom zvolené toto nastavenie parametru `Overflow`, tak je počas behu rozšírenia vytvorená pomocná kocka, ktorá je potom použitá na orezávanie presahujúcich objektov. Orezávanie funguje na prin-



Obr. 4.9: Príklad nastavení parametru Inherit Rotation, ktorého cieľom je aplikovať rotácie vzniknuté počas otáčania polygónu pri príprave na objekty. Zľava po riadkoch - None (žiadne dedenie rotácie), Simple (dedenie rotácie podľa osi Z), Complex (dedenie rotácie podľa ôs X a Y), Full (dedenie rotácii na všetkých troch osách).

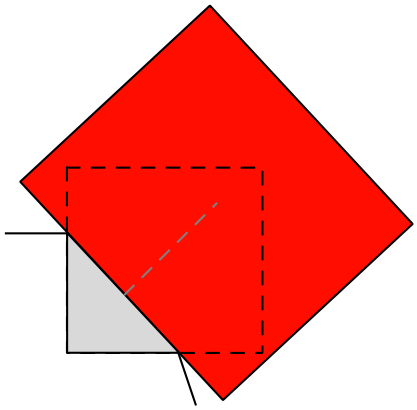
cípe nájdenia dvoch bodov, kde sa pretínajú hranice vygenerovaného objektu a pokrývanej plochy (obrázok 4.10).

Pomocou týchto dvoch bodov sa vypočíta pozícia, na ktorú je potrebné pomocnú kocku posunúť, aby dokázala správne orezať presahujúci objekt. Toto nastavenia parametru umožňuje len jednoduché orezávanie objektov. V prípade zložitejšej geometrie časti plochy, cez ktorú objekt presahuje, sa vytvorí len jednoduché orezanie podľa dvoch bodov, ktoré nebude presne sledovať hrany plochy (obrázok 4.11).

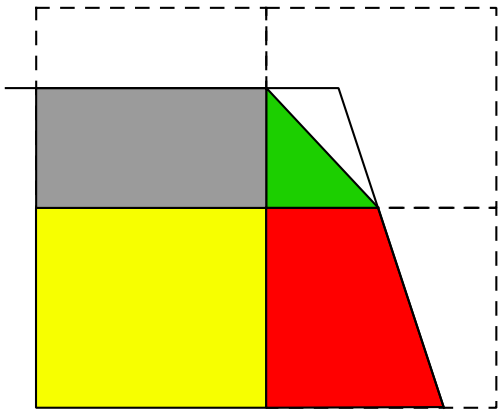
Nastavenie `Full` ponecháva presahujúce objekty a nijakým spôsobom ich nenahradzuje alebo nemodifikuje.

4.4.6 Link Meshes

Tento parameter umožňuje ponechať prepojenie geometrie vytvorených objektov s ich vzorom. To umožňuje užívateľovi modifikovať geometriu všetkých kópii pôvodného objektu naraz. V prípade, že parameter `Overflow` je nastavený na možnosť `Sliced`, nie je možné použiť tento parameter, keďže nastavenie `Sliced` modifikuje geometriu presahujúcich objektov.



Obr. 4.10: Obrázok zobrazuje spôsob, akým sa pomocou kocky (červený objekt) orezáva do plochy vložený objekt. Pozícia kocky je vypočítaná pomocou jej rozmerov a hrany, ktorá vznikne spojením dvoch bodov pretnutia vkladaného objektu s hranami plochy.



Obr. 4.11: Príklad parametru Overflow s nastavením Sliced. V tomto nastavení parameter odreže presahujúce časti objektov pomocou modifikátoru Boolean. Vytvára sa len jednoduché orezávanie objektov, preto v prípade presahovanie trochu bodov nekopíruje orezaný objekt hrany polygónu (zelený objekt).

Kapitola 5

Meranie

Po dokončení implementácie rozšírenia bolo vykonané testovanie za účelom zistenia časovej náročnosti behu rozšírenia. Toto testovanie bolo rozdelené do dvoch častí. Prvá časť sa zaoberá vplyvom množstva vygenerovaných objektov na dĺžku behu rozšírenia. Druhá časť testovania sa zaoberá vplyvom jednotlivých užívateľských parametrov na dĺžku behu v prípade, že ostatné vstupné podmienky sú nemenné.

Testovanie sa vykonávalo ako rozdiel systémového času na začiatku a na konci behu operátoru. Každý test bol vykonaný desať krát a výsledný čas bol vypočítaný ako priemerná hodnota zo všetkých behov. Počas každého behu boli odstraňované objekty vzniknuté počas predchádzajúceho behu. Pred prvým meraným behom sa vykonal jeden nemeraný beh, ktorý zabezpečil, že prvý beh odstraňoval rovnaké množstvo vygenerovaných objektov ako všetky nasledujúce behy.

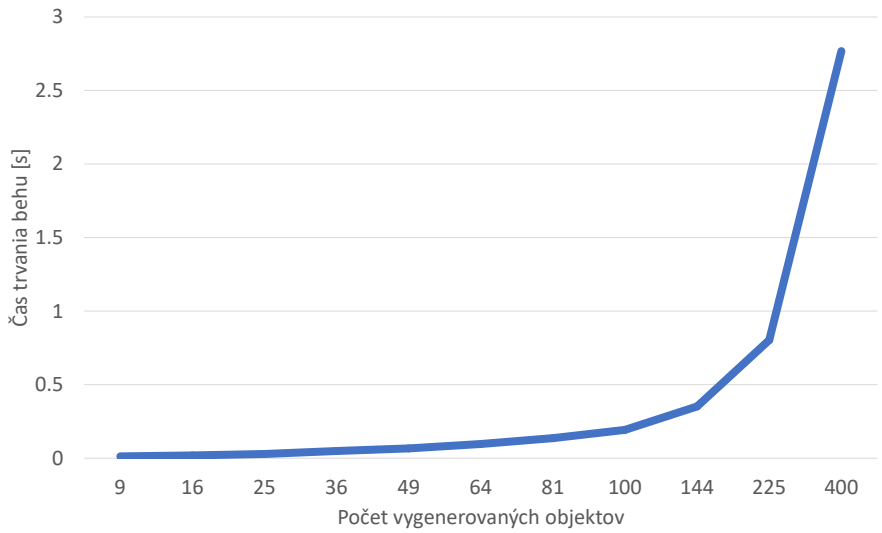
5.1 Vplyv veľkosti výstupu

Počet vygenerovaných objektov	Čas [s]
9	0.01270
16	0.01885
25	0.02891
36	0.04892
49	0.06687
64	0.09664
81	0.13659
100	0.19282
144	0.35203
225	0.80323
400	2.76820

Tabuľka 5.1: Tabuľka zobrazujúca vplyv množstva vygenerovaných objektov na dĺžku behu. Čas trvania každého behu je vypočítaný ako priemerná hodnota z desiatich behov s rovnakými počiatočnými parametrami. Táto hodnota je zaokrúhlená na päť desatinných čísel.

Testovanie vplyvu množstva vygenerovaných objektov na dĺžku behu rozšírenia (tabuľka 5.1, graf 5.1) bolo vykonávané pomocou jednoduchej scény, kde sa na pokrývanie používala

kolekcia obsahujúca len jeden objekt - základnú kocku. Pokrývaná bola len jedna plocha, ktorá sa po úpravách na začiatku behu skladala len z jedného polygónu štvorcového tvaru. Počas testovania sa menila len veľkosť pokrývanej plochy. Účelom tejto zmeny bolo vygenerovanie výsledkov s rôznym počtom objektov. Pri tomto testovaní neboli použité žiadne užívateľské parametre.



Obr. 5.1: Graf zobrazujúci závislosť dĺžky behu rozšírenia na počte vygenerovaných objektov. Dĺžka behu je vypočítaná ako priemerná hodnota z desiatich behov.

Z výsledkov testovania je možné si povšimnúť, že dĺžka behu rozšírenia nerastie lineárne s počtom vygenerovaných objektov ale skôr exponenciálne. Príčinou toho je možné získať výsledky behu v reálnom čase len pri menšom počte objektov. V prípade, že výsledkom behu rozšírenia sú stovky, prípadne tisícky objektov, tak výsledky behu vzniknú až s niekoľko sekundovým oneskorením.

5.2 Vplyv parametrov

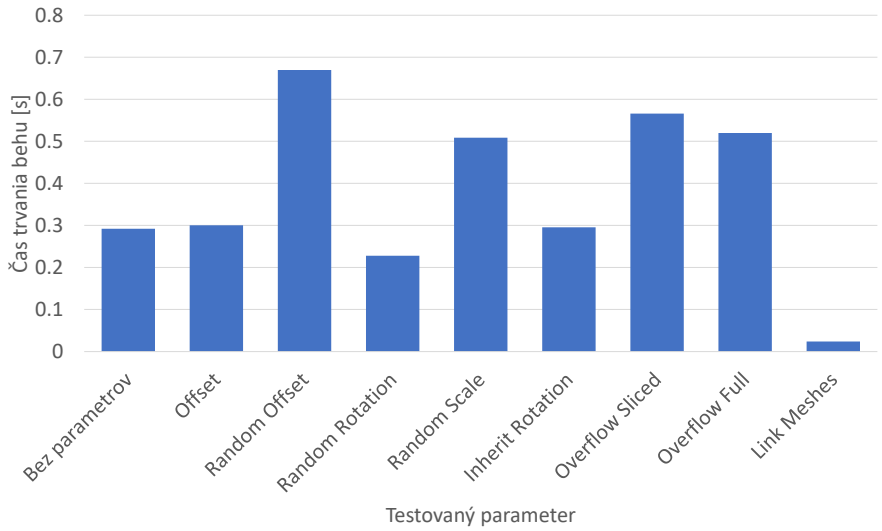
Táto časť testovania sa zameriavala na pozorovanie vplyvu užívateľských parametrov na dĺžku behu. Oproti predchádzajúcemu testovaniu bol do kolekcie pokrývajúcich objektov pridaný druhý objekt - kocka polovičnej veľkosti. Táto zmena bola vykonaná za účelom lepšieho využitia užívateľského parametru `overflow`, kedy bude možné nahradzovať presahujúce objekty a nie ich len vymazávať. Ďalšia zmena sa týkala pokrývanej oblasti. Jej tvar bol zmenený na nepravidelný päťuholník, aby umožňoval vytvorenie presahujúcich objektov, čo pri ploche štvorcového tvaru nie je možné.

Počas jednotlivých behov sa vstupné objekty nemenili. Jediné zmeny, ktoré nastávali bolo aktivovanie a deaktivovanie jednotlivých užívateľských parametrov. Počas merania bol zaznamenávaný aj počet vygenerovaných objektov, keďže jednotlivé parametre priamo ovplyvňujú počet objektov, ktorý bude počas behu vytvorený. Aby sa minimalizoval rozdiel vygenerovaných objektov, tak boli do parametrov vložené také hodnoty, ktoré sa čo najviac približovali hodnotám využívaných v prípade, že by neboli parametre aktivované.

Parameter	Čas [s]	Počet vygenerovaných objektov
Bez parametrov	0.29201	104
Offset	0.30019	102
Random Offset	0.66967	105
Random Rotation	0.22788	85
Random Scale	0.50851	94
Inherit Rotation - Full	0.29537	104
Overflow Sliced	0.56586	158
Overflow Full	0.51971	158
Link Meshes	0.02369	104

Tabuľka 5.2: Tabuľka zobrazujúca vplyv parametrov na čas behu a množstvo vygenerovaných objektov. Dĺžka behu je vypočítaná ako priemerná hodnota z desiatich behov s rovnakými počiatočnými parametrami. Táto hodnota je zaokrúhlená na päť desatinných čísel. Rozdielne počty objektov sú spôsobené tým, že niektoré parametre priamo vplývajú na množstvo objektov, ktoré sa do plochy zmestia, prípadne umožňujú ponechanie presahujúcich objektov.

Z výsledkov testovanie (tabuľka 5.2, graf 5.2) je možné si povšimnúť, že väčšie rozdiely v dĺžke behov spôsobujú parametre, ktoré využívajú pre svoju funkcionalitu určitú náhodnosť alebo upravujú geometriu objektov. Avšak najväčší rozdiel spôsobil parameter `link meshes`, pri ktorého aktivovaní nie je potrebné kopírovať pri každom vzniknutom objekte jeho geometriu, čo výrazne zrýchľuje beh rozšírenia. Najmenší vplyv na dĺžku behu mal parameter `inherit rotation`, ktorý len mení rotáciu vygenerovaného objektu na základe rotácii, ktoré sa vypočítavajú pri otáčaní polygónu.



Obr. 5.2: Graf zobrazujúci závislosť dĺžky behu rozšírenia na použití užívateľských parametrov. Dĺžka behu je vypočítaná ako priemerná hodnota z desiatich behov.

Kapitola 6

Záver

Výsledkom tejto práce je rozšírenie, ktoré sa zaoberá pokrývaním plôch užívateľom vytvorenými objektmi pomocou binárneho stromu. Rozšírenie je implementované pomocou jazyka Python v spojení s rozhraním Blender API. Jeho cieľom je uľahčiť prácu pri tvorbe rozsiahlych scén. V aktuálnej implementácii rozšírenie umožňuje užívateľom pokrývať plochy objektmi, nastavovať rôzne parametre pre tieto objekty, ako ich veľkosť, otočenie a vzdialenosť medzi nimi. Do tejto vzdialenosti je možné vložiť aj záporné hodnoty, pri ktorých sa budú jednotlivé vygenerované objekty prekrývať. Ďalej je možné pomocou parametrov určovať, ako sa bude rozšírenie vysporiadávať s objektmi, ktoré presahujú cez hranice pokrývanej plochy.

Vďaka tomu, že nie je obmedzený počet, či typ objektov, ktoré sú použité na pokrývanie je toto rozšírenie veľmi všestranné. Je možné ho využiť na vytváranie rozličných veľkých oblastí v scénach, ako napríklad oblastí prírody alebo mestských oblastí. Zároveň môže byť využité aj na vytvorenie menších, detailnejších objektov ako napríklad oplotení, chodníkov alebo rôznych kamenných štruktúr.

Z vykonaných meraní je možné si všimnúť, že rozšírenie funguje dostatočne rýchlo pri generovaní menšieho množstva objektov. V tomto prípade výrazným spôsobom negatívne neovplyvňujú prácu s rozšírením ani časovo náročnejšie parametre. Avšak s väčším počtom objektov sa dĺžka behu rozšírenia výrazne zvyšuje. Z toho dôvodu by bolo vhodným rozšírením do budúcnosti preskúmať možnosť zefektívnenia kódu, aby bolo možné generovať stovky, prípadne tisícky objektov bez čakania. Ďalším rozšírením, ktoré by bolo vhodné implementovať, je komplexné orezávanie objektov, ktoré by dokázalo kopírovať hrany pokrývanej plochy. Vďaka tomu by vznikalo menej nevyplneného priestoru pri hranách polygónu, ktoré sú spôsobené aktuálnym jednoduchým orezávaním.

Literatúra

[1] Almeida Cunha, T. d. Q. J.G. de. Grids for cutting and packing problems: a study in the 2D knapsack problem. *4OR*. 2020, s. 293 – 339, [cit. 2023-02-19]. DOI: https://doi.org/10.1007/s10288-019-00419-9. Dostupné z: https://link.springer.com/article/10.1007/s10288-019-00419-9?utm_source=getftr&utm_medium=getftr&utm_campaign=getftr_pilot.

[2] *Blender 3.4 Python API Documentation* [online]. Blender Foundation, 2022 [cit. 2022-12-27]. Dostupné z: https://docs.blender.org/api/current/index.html.

[3] Bortfeldt, A. a Winter, T. A genetic algorithm for the two-dimensional knapsack problem with rectangular pieces. *International Transactions in Operational Research*. November 2009, zv. 16, s. 685 – 713, [cit. 2023-02-19]. DOI: 10.1111/j.1475-3995.2009.00701.x.

[4] Chekanin, A. a Chekanin, V. Efficient algorithms for orthogonal packing problems. *Computational Mathematics and Mathematical Physics*. Október 2013, zv. 53, s. 1457–1465, [cit. 2023-02-19]. DOI: 10.1134/S0965542513100047.

[5] Chen, P., Fu, Z., Lim, A. a Rodrigues, B. Two-dimensional packing for irregular shaped objects. In:. Február 2003, s. 10 pp. [cit. 2023-01-15]. DOI: 10.1109/HICSS.2003.1174211. Dostupné z: https://www.researchgate.net/publication/224739203_Two-dimensional_packing_for_irregular_shaped_objects.

[6] Cid Garcia, N. M. a Rios Solis, Y. A. Positions and covering: A two-stage methodology to obtain optimal solutions for the 2d-bin packing problem. *PLOS ONE*. Public Library of Science. Apríl 2020, zv. 15, č. 4, s. 1–22, [cit. 2023-02-19]. DOI: 10.1371/journal.pone.0229358. Dostupné z: https://doi.org/10.1371/journal.pone.0229358.

[7] Cid Garcia, N. M. a Rios Solis, Y. A. Exact solutions for the 2d-strip packing problem using the positions-and-covering methodology. *PLOS ONE*. Public Library of Science. Január 2021, zv. 16, č. 1, s. 1–20, [cit. 2023-02-19]. DOI: 10.1371/journal.pone.0245267. Dostupné z: https://doi.org/10.1371/journal.pone.0245267.

[8] Clautiaux, F., Carlier, J. a Moukrim, A. A new exact method for the two-dimensional orthogonal packing problem. *European Journal of Operational Research*. 2007, zv. 183, č. 3, s. 1196–1211, [cit. 2023-02-19]. DOI: https://doi.org/10.1016/j.ejor.2005.12.048. ISSN 0377-2217. Dostupné z: https://www.sciencedirect.com/science/article/pii/S0377221706002992.

[9] Dinas, J. M. A literature review of bounding volumes hierarchy focused on collision detection. *Ingeniería y Competitividad*. 2015, [cit. 2023-01-16]. ISSN 0123-3033. Dostupné z: https://www.redalyc.org/articulo.oa?id=291339265004.

[10] Eberly, D. H. *3D Game Engine Design: A Practical Approach to Real-TimeComputer Graphics*. Second. San Francisco: Morgan Kaufmann Publishers, 2007 [cit. 2023-01-22]. ISBN 1-55860-593-2.

[11] Guo, B., Zhang, Y., Hu, J., Li, J., Wu, F. et al. Two-dimensional irregular packing problems: A review. *Frontiers in Mechanical Engineering*. 2022, zv. 8, [cit. 2023-01-15]. DOI: 10.3389/fmech.2022.966691. ISSN 2297-3079. Dostupné z: https://www.frontiersin.org/articles/10.3389/fmech.2022.966691.

[12] Kennedy, T. *Monte Carlo Methods - a special topics course* [Lecture for Monte Carlo Methods - Course]. 2016 [cit. 2023-01-08]. Dostupné z: https://www.math.arizona.edu/~tgk/mc/book.pdf.

[13] Liu, Z., Huang, M. a Zhu, S. The Design and Implementation of a Pseudo Random Number Generation Algorithm. In: IEEE. *2009 International Conference on Computational Intelligence and Natural Computing*. 2009, sv. 2, s. 126–129. DOI: 10.1109/CINC.2009.242. ISBN 978-0-7695-3645-3.

[14] Meng, X. *Linear Congruential Method* [online]. 2002 [cit. 2023-01-22]. Dostupné z: https://www.eg.bucknell.edu/~xmeng/Course/CS6337/Note/master/node40.html.

[15] Oliveira Óscar, S. E. An introduction to the two-dimensional rectangular cutting and packing problem. *International Transactions in Operational Research*. 2022, [cit. 2023-01-15]. DOI: 10.1111/itor.13236. Dostupné z: https://onlinelibrary.wiley.com/doi/full/10.1111/itor.13236.

[16] Sidhpurwala, H. *Understanding random number generators, and their limitations, in Linux* [online]. RED HAT BLOG, 2022 [cit. 2023-01-22]. Dostupné z: https://www.redhat.com/en/blog/understanding-random-number-generators-and-their-limitations-linux.

[17] Sigman, K. *Random Number Generators* [online]. 2019 [cit. 2023-01-22]. Dostupné z: http://www.columbia.edu/~ks20/4106-18-Fall/Simulation-LCG.pdf.

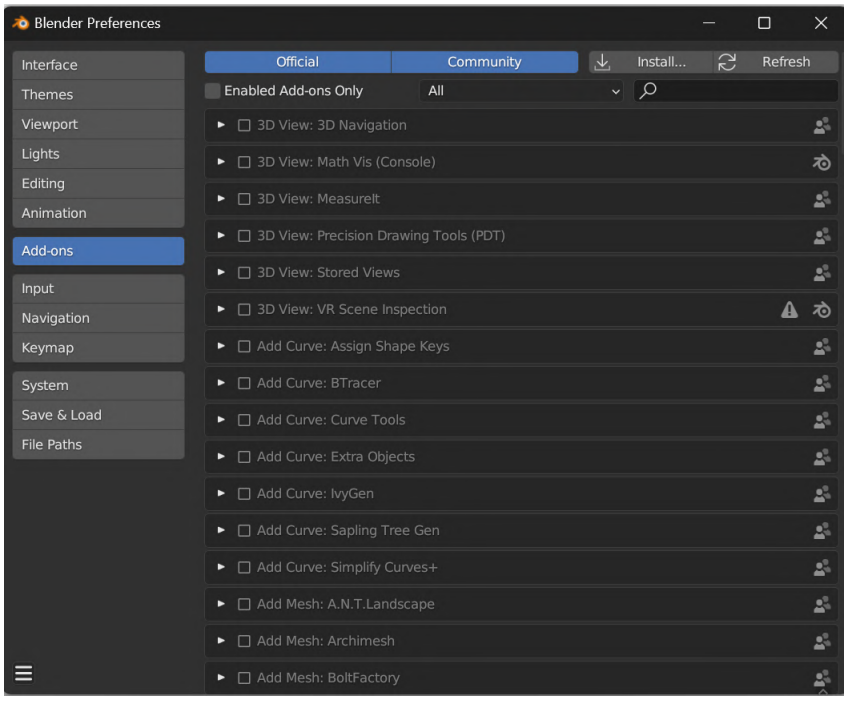
Príloha A

Inštalácia rozšírenia

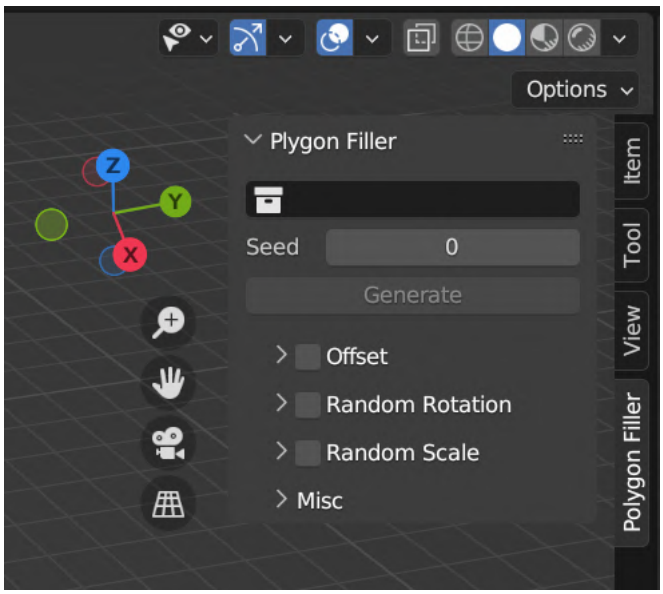
Rozšírenie je dostupné k stiahnutiu na adrese: https://jakubjsmykal.gumroad.com/l/pfill

Po stiahnutí rozšírenia, ktoré je uložené vo forme .zip archívu, je možné toto rozšírenie nainštalovať cez aplikáciu Blender. Pre inštaláciu je potrebné otvoriť Blender a jeho nastavenia, ku ktorým sa dá dostať pomocou cesty `Edit -> Preferences -> Add-ons` (obrázok A.1). Následne je potrebné stlačiť tlačidlo `install` a presunúť sa cez novo otvorené okno do priečinku, kam bolo počas sťahovania rozšírenie uložené. Stlačením tlačidla `Install Add-on` sa nainštaluje rozšírenie, ale neaktivuje sa. Je potrebné ho nájsť v zozname dostupných rozšírení a aktivovať ho prepnutím checkboxu.

Užívateľské prostredie rozšírenia je možné nájsť v záložke `Polygon Filler` (obrázok A.2).



Obr. A.1: Nastavenia aplikácie Blender, pomocou ktorých sa dajú inštalovať rozšírenia.

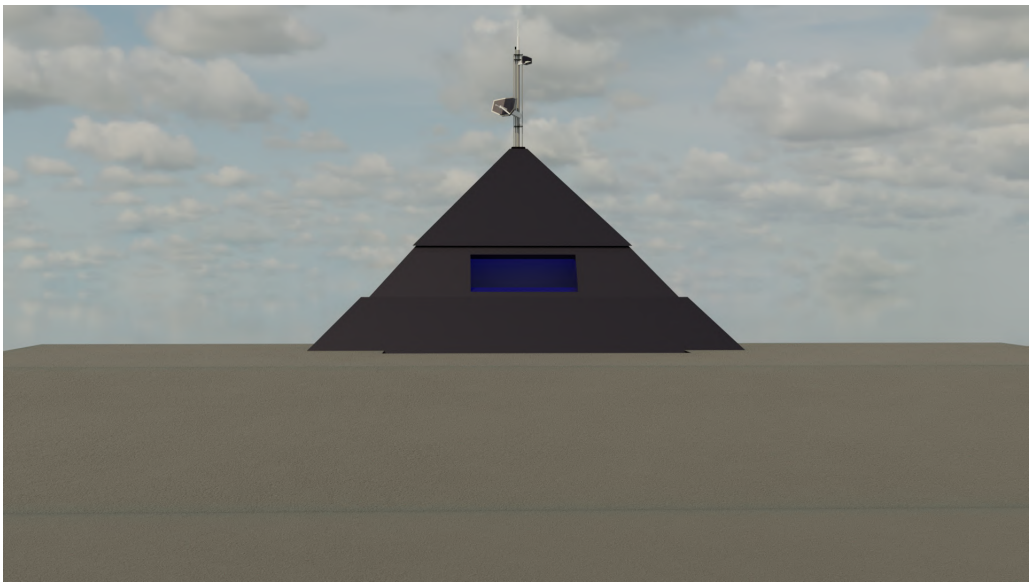


Obr. A.2: Umiestnenie ovládania rozšírenia v rámci užívateľského prostredia aplikácie Blender.

Príloha B

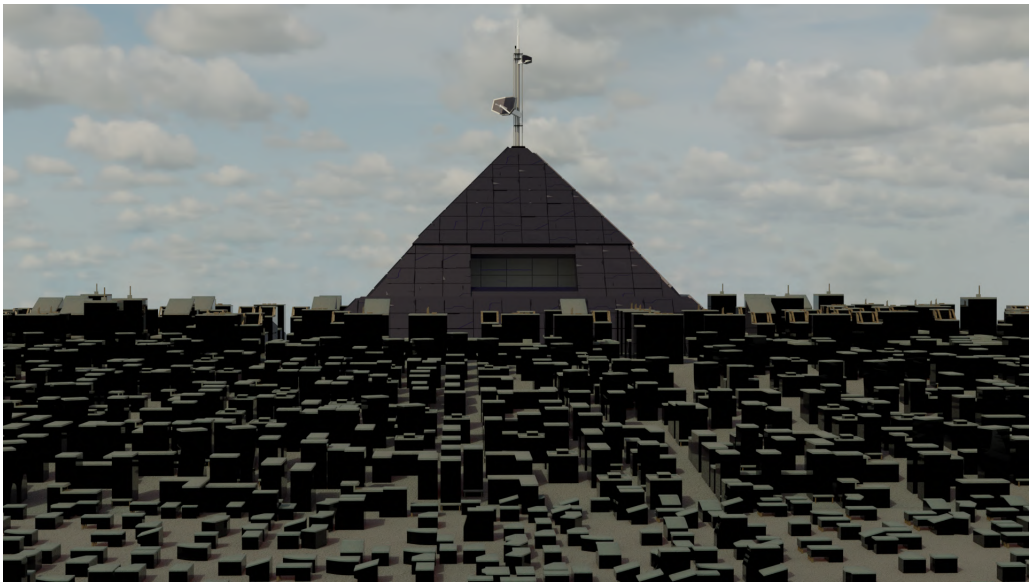
Príklady využitia rozšírenia

Príklady scén vytvorených pomocou rozšírenia `PolygonFiller`, ktoré je výsledkom tejto práce. V prípade prvej scény (obrázok B.1, obrázok B.2) bolo z internetu prebraté pozadie a textúry použité na zemi a budovách. Modely boli vytvorené. Pri druhej scéne (obrázok B.3, obrázok B.4) boli z internetu prebraté modely rastlín a kameňov. Ďalej boli prebraté všetky použité textúry. Všetky modely a textúry, ktoré boli prebraté sú na internete dostupné pod licenciou CC0[1].



Obr. B.1: Príklad scény zobrazujúcej mesto. Táto scéna bola použitá ako základ, ktorý sa potom doplní pomocou rozšírenia PolygonFiller.

[1] https://polyhaven.com



Obr. B.2: Príklad scény zobrazujúcej mesto, ktorá bola vytvorená pomocou rozšírenia PolygonFiller.

Obr. B.3: Príklad scény zobrazujúcej rieku a okolitú prírodu. Táto scéna bola použitá ako základ, ktorý sa potom doplní pomocou rozšírenia PolygonFiller.

Obr. B.4: Príklad scény zobrazujúcej rieku a okolitú prírodu, ktorá bola vytvorená pomocou rozšírenia PolygonFiller.