

VILNIAUS GEDIMINO TECHNIKOS UNIVERSITETAS

FUNDAMENTINIŲ MOKSLŲ FAKULTETAS

INFORMACINIŲ SISTEMŲ KATEDRA

Kasparas Kažemėkas

**Procedūrinis pastatų generavimas naudojant UNITY 3D**

**Procedural Buildings Generation Using UNITY 3D**

Baigiamasis bakalauro darbas

Programų sistemų inžinerijos studijų programa, valstybinis kodas 612I30003

Programų sistemų studijų kryptis

Vilnius, 2019

VILNIAUS GEDIMINO TECHNIKOS UNIVERSITETAS

FUNDAMENTINIŲ MOKSLŲ FAKULTETAS

INFORMACINIŲ SISTEMŲ KATEDRA

TVIRTINU

Katedros vedėjas

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Parašas)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Vardas, pavardė)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Data)

Kasparas Kažemėkas

**Procedūrinis pastatų generavimas naudojant UNITY 3D**

**Procedural Buildings Generation Using UNITY 3D**

Baigiamasis bakalauro darbas

Programų sistemų inžinerijos studijų programa, valstybinis kodas 612I30003

Programų sistemų studijų kryptis

**Vadovas**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_

(Pedag. vardas, vardas, pavardė) (Parašas) (Data)

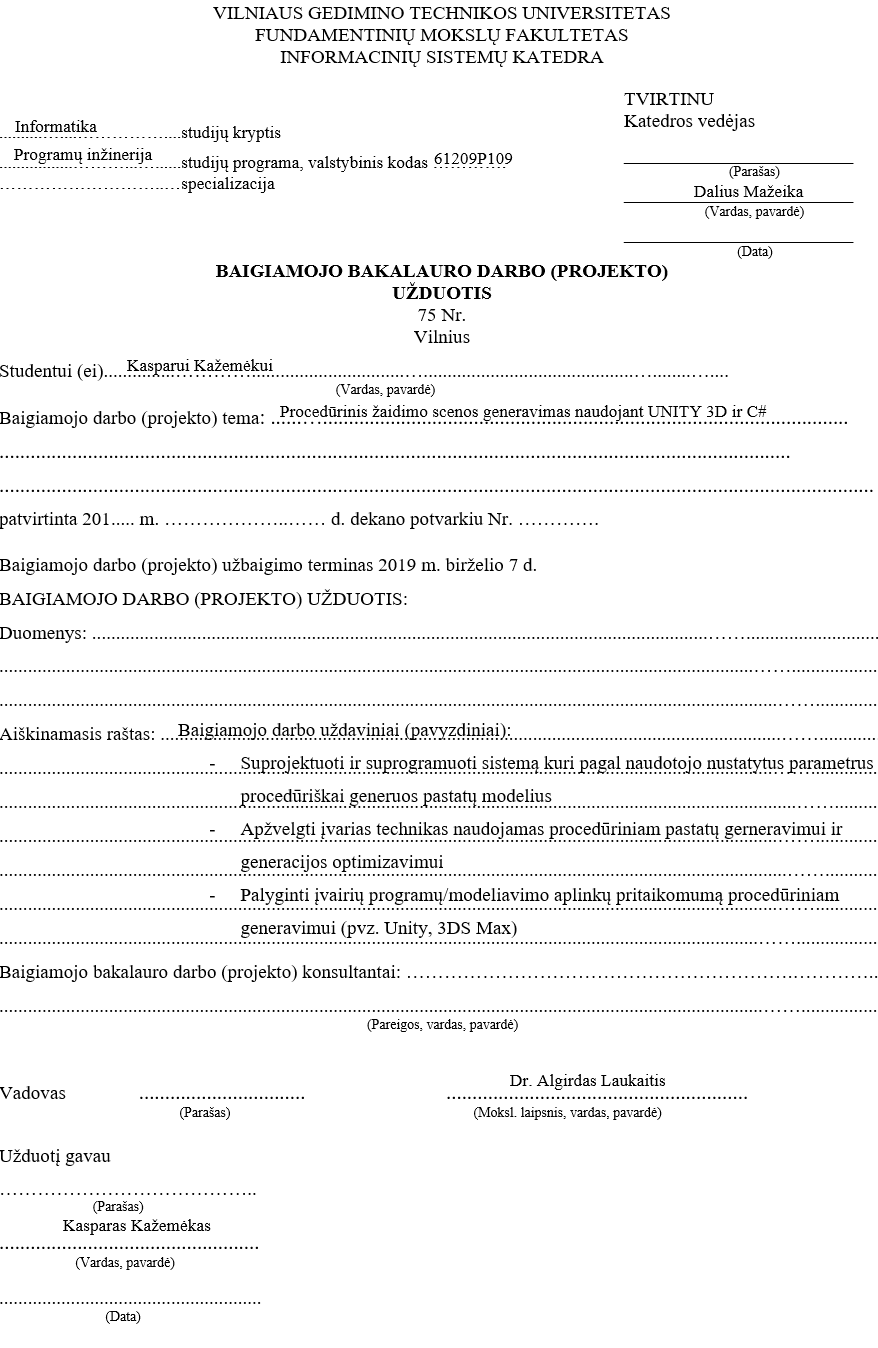
**Konsultantas**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_

(Pedag. vardas, vardas, pavardė) (Parašas) (Data)

**Konsultantas**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_

(Pedag. vardas, vardas, pavardė) (Parašas) (Data)

Vilnius, 2019



**Anotacija**

Šioje medžiagoje aprašoma bakalaurų baigiamųjų darbų paskirtis ir turinys, pateikiami bakalaurų baigiamųjų darbų rengimo patarimai, pavyzdžiai ir įforminimo taisyklės.

Be to ši medžiaga gali būti naudojama, kaip šablonas, rengiant baigiamąjį bakalauro darbą, pradedant nuo turinio ir baigiant literatūros sąrašu. Bakalarų baigiamųjų darbų anotacija lietuvių kalba ir anglų kalba čia nepateikiami. Jos turi būti pildomos ir spausdinamos iš mano.vgtu.lt.

**Turinys**

[1. Įvadas 7](#_Toc28944183)

[Darbo tikslas ir uždaviniai 7](#_Toc28944184)

[2. Procedūrinio modelių generavimo technologijų analizė 8](#_Toc28944185)

[2.1. Viduramžių namo architektūrinė analizė 8](#_Toc28944186)

[2.2. Programų apžvalga 9](#_Toc28944187)

[2.2.1. Houdini Building Generator [5] 9](#_Toc28944188)

[2.2.2. Maya Structures [6] 10](#_Toc28944189)

[2.2.3. Building Generator v0.7 [7] 11](#_Toc28944190)

[2.2.4. BuildR 2 [8] 11](#_Toc28944191)

[2.2.5. SceneCity [9] 12](#_Toc28944192)

[2.3. Procedūrinio generavimo algoritmų apžvalga 13](#_Toc28944193)

[2.3.1. L-sistema ir jos plėtiniai 13](#_Toc28944194)

[2.3.2. Greuter metodas 14](#_Toc28944195)

[2.3.3. Momentinė architektūra 15](#_Toc28944196)

[2.4. Skyriaus išvados 17](#_Toc28944197)

[3. Projektinė dalis 17](#_Toc28944198)

[3.1. Rekalavimų specifikacija 17](#_Toc28944199)

[3.1.1. Formuluojamos užduotys 17](#_Toc28944200)

[3.1.2. Funkciniai sistemos reikalavimai 17](#_Toc28944201)

[3.1.3. Nefunkciniai reikalavimai 17](#_Toc28944202)

[3.2. Programų sistemos projektas 17](#_Toc28944203)

[3.2.1. Programų sistemos projektiniai reikalavimai 17](#_Toc28944204)

[3.2.2. Programų sistemos architektūra 17](#_Toc28944205)

[3.2.3. Programų sistemos maketai 17](#_Toc28944206)

[4. Testavimas 18](#_Toc28944207)

[5. Išvados ir siūlymai 18](#_Toc28944208)

[Literatūra ir šaltiniai 19](#_Toc28944209)

Santrumpų ir terminų žodynas

Verteksas – 3D objekto taškas erdvėje, viršūnė.

Mesh – verteksų tinklas sudarantis 3D objekto formą

Poligonas – Plokštuma gaunama kelis verteksus sujungus kraštinėmis (pavyzdžiui trikampis)

Modelis – 3D objektas.

LOD (level of detail) – modelio detalumas, kuo objektas toliau nuo stebėtojo, tuo LOD gali būti mažesnis.

Low poly – 3D grafikos stilius kai kuriami objektai, dėl estetinių ar techninių prižasčių, turi salyginai mažą kiekį poligonų.

# Įvadas

Žaidimų industrija 2019 metais gavo 152 milijardus dolerių pajamų, per metus paaugdama 9.6% [1]. Daugiau nei trečdalį pajamų atnešė mobiliems įrenginiams skirti žaidimai. Tai svarbu, kadangi kiekvienam rinkos segmentui naudojama skirtingo pajėgumo techninė įrangą, o silpniausių įrenginių dominavimas parodo, kad daugelis žaidimų, lyginant su asmeniniams kompiuteriams skirtais žaidimais, yra grafiškai paprasti. Prie mobilių telefonų žaidimo grafinio paprastumo prisideda ir visose žaidimų platformose populiarius low poly stilius [2] (geri jo pavyzdžiai būtų „Monument Valley“ ir „Astroneer“). Žinoma grafinis paprastumas nereiškia, kad žaidimo aplinką lengvą sukurti, atvirkščiai, maži poligonų limitai reikalauja kokybiškos optimizacijos ir sunkiai palaikomo balanso tarp detalumo ir veikimo greičio.

Žaidimų pardavimo kaina jau dešimtmetį išlieka ta pati – $60 [3], atsižvelgiant į infliaciją, akivaizdu, kad kasmet ta pati kaina duoda vis mažesnę vertę žaidimų kūrėjams, taigi jiems tenka, be kitų pelno auginimo priemonių, pastoviai optimizuoti žaidimų kūrimo procesą. Procedūrinis žaidimo aplinkos generavimas – vienas iš kaštų mažinimo būdų. Procedūriškai generuoti objektai gali būti naudojami greitam prototipavimui – kurti bazinę žaidimo aplinkos išvaizdą, vėliau procedūrinius modelius pakeičiant kurtais rankomis. Tačiau vis tobulėjanti techninė įranga leidžia lengvai kurti ir kokybiškus galutiniame produkte naudojamus objektus ar net generuoti ištisą visatą su unikaliomis planetomis turinčiomis savo ekosistemas (No Man‘s Sky).

Paprastai procedūriniame modelių generavime naudojami iš anksto paruošti bazinių modelių rinkiniai (tarkim durys, sienos, langai, stogo segmentai) turintys ir iš anksto sukurtas tekstūras. Rinkiniai panaudojami procedūriškai surinkti objektus. To pliusas, kad generuoti objektai gali būti geometriškai sudėtingi ir itin realistiški. Iš kitos pusės, bazinių modelių kūrimas reikalauja nemažai laiko, o juo sukūrus vėliau gali būti sudėtinga atlikti pakeitimus. Tokios generacijos pakaitalas – procedūrinis visų objekto dalių generavimas nuo pradžios iki pabaigos įskaitant ir tekstūras, ko ir bus siekiama kuriant šią sistema.

Šiuo metu siūlomi pastatų ir miestų generatoriai arba orientuoti į realistinį šiuolaikinių pastatų generavimą bei universalumą. Akivaizdu, kad universalūs įrankiai primtini plačiam vartotojų ratui, o griežtos modernių pastatų formos leidžia nesunkiai, pasinaudojus tomis pačioms taisyklėmis, generuoti platų spektrą pastatų. Tačiau, nemažo skaičiaus žaidimų veiksmas vyksta viduramžių ar fantastinėse aplinkose, kur tokie generatoriai sunkiai pritaikomi.

Taigi, atsižvelgus į mobilių įrenginių dominavimą, low poly grafinio stiliaus populiarumą ir į šiuolaikinę architetūra orientuotus generatorius iškyla poreikis įrankiui leidžiančiam generuoti labiau organinius, stilizuotus viduramžių pastatus.

Darbo tikslas ir uždaviniai:

**Darbo tikslas –** apžvelgti egzistuojančių pastatų generatorių galimybes, išsiaiškinti galimus pastatų generavimo metodus bei jų taikymą. Pasirinkti tinkamiausius metodus kuriamam low poly viduramžių pastatų generatoriui.

**Tyrimo objektas** – automatiniai virtualių pastatų generavimo metodai.

**Tyrimo problema** – Pastatų generatorių, orientuotų į stilizuotą viduramžišką stilių, trūkumas.

**Darbo uždaviniai:**

1. Apžvelgti rinkoje egzistuojančių pastatų generatorių galimybes;
2. Išnagrinėti populiariausius procedūrinių 3D objektų generavimo metodus;
3. Atlikti pastatų generatoriaus projektavimą, pasirinkti tinkamą programinę įrangą;
4. Atrinkti geriausiai low poly viduramžių pastatų generavimui tinkamus metodus;
5. Realizuoti programą, atlikti bandymus, suformuluoti išvadas;

**Darbo struktūra**:

* Antras skyrius – egzistuojančių pastatų generatorių apžvalga bei pastatų generavimo algoritmų analizė
* Trečias skyrius – projektinė dalis
* Ketvirtas skyrius – testavimas
* Penktas skyrius - išvados

# Procedūrinio modelių generavimo technologijų analizė

## Viduramžių namo architektūrinė analizė

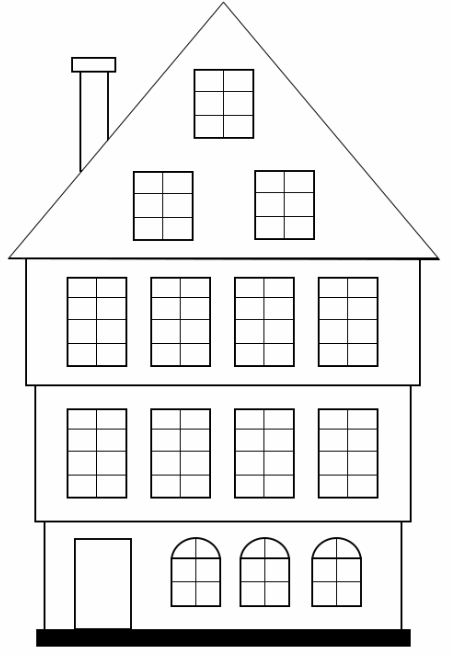
Kuriama sistema orientuosis į viduramžių namų generavimą, tad visų pirma reikia apibrėžti pagrindines tokių namų savybes. Kaip lengvai atpažystamas pavyzdys, savotiškas viduramžių miesto pastato etalonas, pasirinktas vokiškas fachverko stiliaus namas (1 pav.).

* Matoma, kad siekiant kuo efektyviau išnaudoti mažą žemės sklypa, kiekvienas namo aukštas būdavo statomas vis labiau išsikišęs į gatvę.
* Tokie namai turi stačius, šlaitinius, čerpių stogus, palėpėje sutalpinant po porą aukštų.
* Dominuoja auškti, tankiai sudėti segmentuoti langai.
* Langai bei durys dviejų formų – arkiniai arba stačiakampiai
* Namai nesimetriškai, sienos kreivokos.
* Pirmo aukšto išvaizda skiriasi nuo sekančių aukštų.
* Vidutiniškai 5 aukštai – 3 pagrindiniai ir 2 palėpėje
* Namai turi kaminus
* Fasadai skirtingų spalvų



Pav. 1 Fachverkiniai namai [4]

Taigi, iš analizės galima sudaryti tokį generuojamo namo planą:



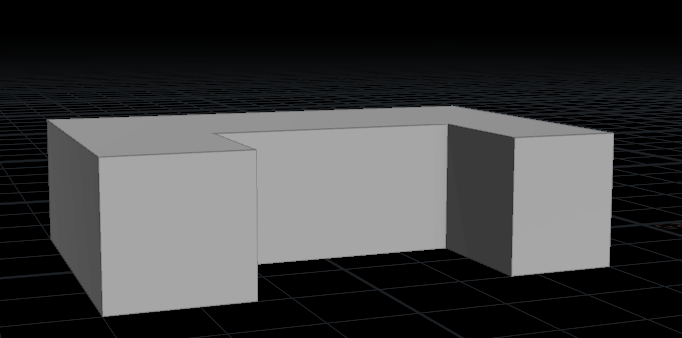
## Programų apžvalga

Šiame poskyriuje atliekama populiarių pastatų/miestų generatorių apžvalga. Kiekvienas įrankis vertinamas pagal jo licencijos tipą, valdymą ir generacijai naudojamus resursus. Taip pat pateikimas kiekvieno įrankio generacijos pavyzdys (žiūrėti priedus). Kiekvienas iš prisatomų įrankių veikia vis kitoje aplinkoje. Apžvelgiamų įrankių sugeneruotų pastatų pavyzdžius galima pamatyti prieduose.

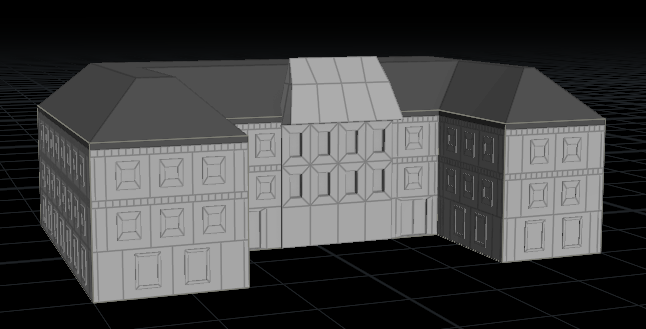
### Houdini Building Generator [5]

|  |  |
| --- | --- |
| Programa/aplinka kurioje veikia generatorius | **Houdini** |
| Naudojimo licenzijos tipas | Patentuota licencija |
| Kaina | Nemokamas su Houdini, tačiau Houdini licencija kainuoja 0-7000$. |
| Programos valdymas | Valdymui naudojamas vizualus programavimas ir nustatymų langai. |
| Veikimo prinicpas | Generacija vykdoma nustatčius pastato tūrį ir jį užpildant panaudojus iš anksto sukurtus komponentus (langus, duris ir t.t.). Komponentai dėliojami taip, kad būtų išlaikytas vieningas ir logiškas pastato stilius, pavyzdžiui durys tik pirmame aukšte ar visame pastato aukšte tos pačios stilistikos ir išmatavimų langai. |
| Panaudojimas | Filmų ir žaidimų aplinkos kūrimas, kompiuterinio meno kūrimas, prototipavimas. |
| Komentarai | Vizualinis programavimas leidžia ne tik gilią įrankio kontrolę, tačiau ir reikalauja minimalių programavimo žinių. |

Lentelė 1 Houdini Building Generator analizė



Pav. 2 Pradinis tūris [5]

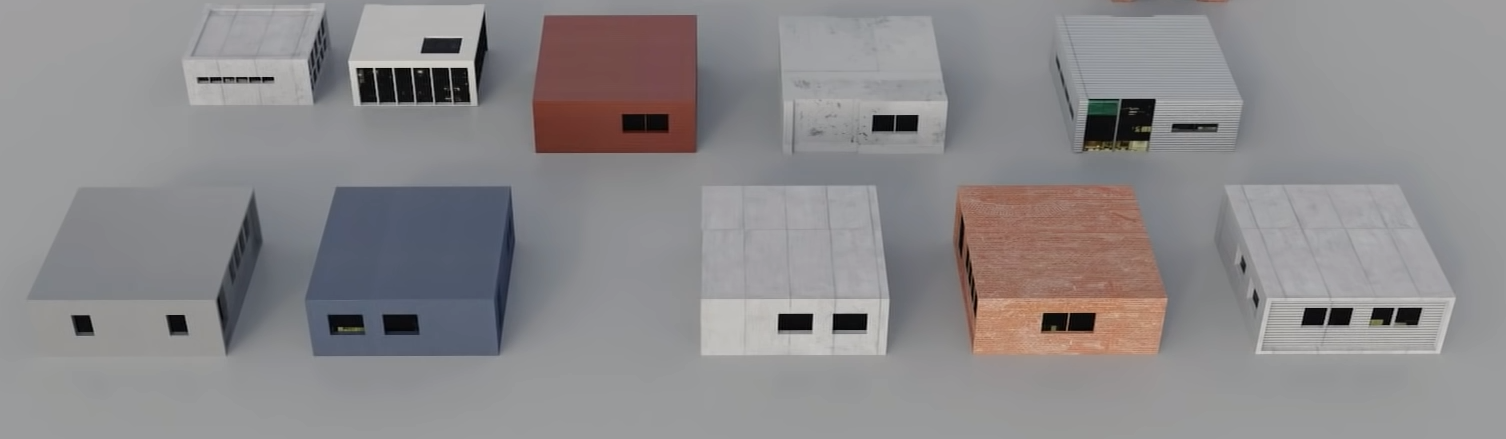


Pav. 3 Sugeneruotas pastatas [5]

### Maya Structures [6]

|  |  |
| --- | --- |
| Programa/aplinka kurioje veikia generatorius | **3Ds Maya** |
| Naudojimo licenzijos tipas | Patentuota licencija |
| Kaina | ~$60 |
| Programos valdymas | Valdymui naudojami tik nustatymų langai. |
| Veikimo prinicpas | Generacija vykdoma panaudojant iš anksto sukurtų modelių rinkinį.Šie modeliai, skirtingai nei kitose pristatomose panašaus veikimo programose, yra ne smulkūs komponentai, kaip langai ar durys, o ištisi pastatų blogai (žr. 5 pav). Programa keisdama modelių geometrines savybes (dydį, pasukimą) ir pozicijas sujungia juos į viena, taip sukurdama skirtingai atrodančius pastatus ir mechanizmus. |
| Panaudojimas | Filmų bei žaidimų aplinkos kūrimas, kompiuterinio meno kūrimas, prototipavimas. |
| Komentarai | Generacija iš pastatų blokų lemia, kad generuojami tik dideli, modernūs ar futuristiniai pastatai ir mechanizmai, kadangi nenaudojant smulkių detalių prarandamas mažiems ar senoviniams pastatams reikalingas detalumas.  Valdymas nustatymų langais leidžia lengviau naudoti programą ir nereikalauja papildomų žinių, kaip vizualinį programavimą naudojantys įrankiai. |

Lentelė 2 Maya Structures pastatų generatoriaus analizė



Pav. 4 Maya Structures generacijai naudojamų blokų pavyzdys [6]

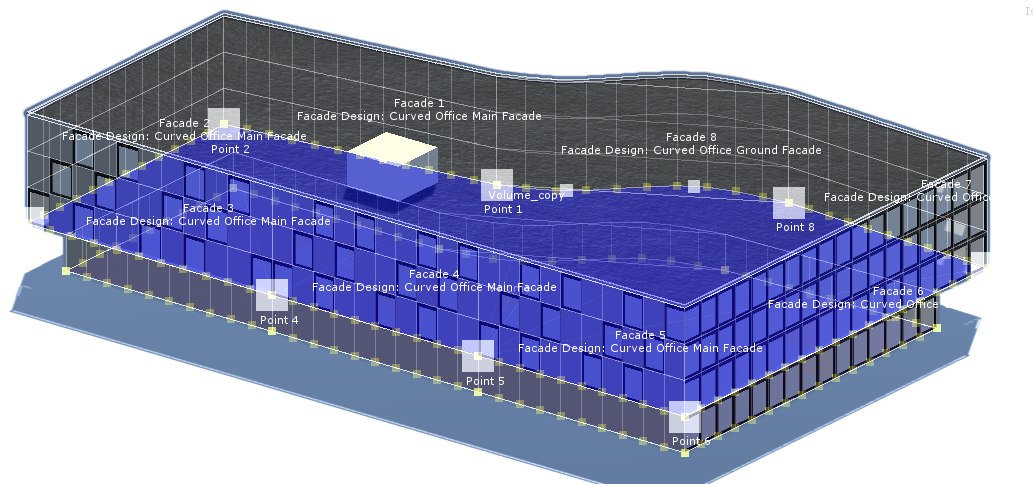
### Building Generator v0.7 [7]

|  |  |
| --- | --- |
| Programa/aplinka kurioje veikia generatorius | **3Ds Max** |
| Naudojimo licenzijos tipas | Atviras Kodas |
| Kaina | Nemokamas |
| Programos valdymas | Valdymui naudojami nustatymų langai |
| Veikimo prinicpas | Generacija, kaip ir Houdini generatoriuje, vykdoma panaudojant iš anksto sukurtų pastato komponentų (langų, durų ir t.t.) rinkinį. Tačiau skirtingai nei Houdini generatoriuje, šiame neįmanoma nustatyti ivarių formų pradinio tūrio, pastatas visuomet yra stačiakampis. Programa keisdama modelių pozicijas ir pasukimo kampą sujungia juos į viena, taip sukurdama skirtingai atrodančius pastatus. Pastatų išvaizda kuriama nustant įvairius parametrus, tokius kaip: aukštų skaičius, balkonų, langų išvaizda, pastato forma ir t.t. |
| Panaudojimas | Filmų bei žaidimų aplinkų kūrimas, kompiuterinio meno kūrimas, prototipavimas. |

Lentelė 3 Building Generator pastatų generatoriaus analizė

### BuildR 2 [8]

|  |  |
| --- | --- |
| Programa/aplinka kurioje veikia generatorius | **Unity** |
| Naudojimo licenzijos tipas | Patentuota licencija |
| Kaina | €88 |
| Programos valdymas | Valdymui naudojami nustatymų langai. |
| Veikimo prinicpas | Įrankis leidžia kurti salyginai primytiviai atrodančius pastatus, kadangi kūrimas susideda iš paprastų 3d modeliavimo operacijų – tokių kaip papildomų verteksų sukūrimas ar tūrio ištraukimas (extract). Procedūrinis šio įrankio aspektas yra automatiškai atliekamas sumodeliuoto objekto skaidymas į segmentus: langus, duris, stogą, sienas, grindis ir jų pavertimas į atitinkamai atrodančius modelius. |
| Panaudojimas | Žaidimų aplinkų kūrimas, prototipavimas |
| Komentarai | Šis įrankis, dėl savo primityvumo, leidžia kurti tik minimalistinius modernius pastatus, kadangi beveik neįmanoma pridėti smulkių detalių.  Objekto skaidymas į segmentus leidžia keisti kiekvieno elemento tekstūras atskirai. |



Pav. 5 BuildR 2 generacijos pavyzdys su matomais verteksais

### SceneCity [9]

|  |  |
| --- | --- |
| Programa/aplinka kurioje veikia generatorius | **Blender** |
| Naudojimo licenzijos tipas | Patentuota licencija |
| Kaina | $97 |
| Programos valdymas | Valdymui naudojamas vizualus programavimas ir nustatymų langai. |
| Veikimo prinicpas | Įrankis leidžia generuoti logiškai išdestytą kelių tinklą, realistiškai atrodantį žemės paviršių ir išdėlioti tūkstančius pastatų. Įrankis gaunamas su paruoštais pastatų pavyzdžiais, taičiau palaiko ir naudotojo sukūrtus pastatus. Pastatai išdėliojami keičiant tik jų pasukimą, bet neliečiant jų formos.  Skirtingai nei kiti apžvelgiami įrankiai, šis neskirtas pilnai proceduriškai generuoti pastatus. Procedūrinio pastatų generavimo galimybės apsiriboja paprastais iš stačiakampių sudėliotais tūriais. Tačiau, siekiant, kad sugeneruoto miesto pastatai neatrodytų vienodi, kiekvienam procedūriniam pastatui sugeneruojamos unikalios tekstūros. Tai pasiekiama „sukarpant“ ir „suklijuojant“ segmentus iš iš anksto paruoštų tekstūrų failų su dideliu kiekiu įvariai atrodančių elementų, pavyzdžiui langų, durų, stogų ir t.t. |
| Panaudojimas | Filmų bei žaidimų aplinkų kūrimas, kompiuterinio meno kūrimas, prototipavimas |
| Komentarai | Įrankis turi Unity žaidimų variklio papildinį leidžiantį sugeneruotus miestus lengvai panaudoti žaidimų kūrime. |

Lentelė 4 SceneCiy mietų generatoriaus analizė

## Procedūrinio generavimo algoritmų apžvalga

### L-sistema ir jos plėtiniai

L-sistema tai septintame praėjusio amžiaus dešimtemetyje vengrų biologo Aristrid Lindenmayerio sukurtas rekursinis modelis aprašantis augalų vystymasi. Dėl sistemos paprastumo ir universalumo ji prigijo ir matematikoje. Sistema tai formali kalba kuri yra aprašoma kaip eilė simbolių, kurių kiekvienas reiškia konkrečią komandą, taip pat sistemoje naudojama eilė parametrų [10]. Kad būtų pasiektas rezultatas, sistema iteruoja per simbolius nustatytą skaičių kartų, kiekvienos iteracijos rezultatui taikydama tas pačias pradžioje aprašytas komandas bei pasinadodamas duodamais parametrais. Pavyzdys [11]:

|  |  |
| --- | --- |
| Simbolis | Reikšmė |
| F | pajudėti per linijos ilgį ir nupiešti liniją |
| - | Pasisukti į kairę per nurodytą kampą |
| + | Pasisukti į dešinę per nurodytą kampą |

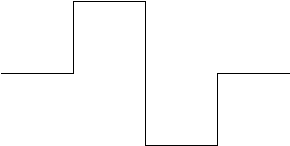
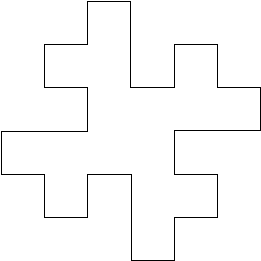
Lentelė 5 simbolių reiškmės

Aksioma = F+F+F+F

F--> F+F-F-FF+F+F-F

Kampas = 90°

Pradinė šio pavydžio simbolių eilutė(aksioma) duoda kvadratą, o pakaitinė eilutė grafiškai pavaizduota 6 pav. Po vienos iteracijos gaunamas 7 pav. vaizdas.

Šis originaliai augalų vystymuisi ir fraktalų aprašymams naudotas modelis gali būti panaudotas ir pastatų generacijai. Pascalis Mülleris bei Yoav I H Parishas savo publikacijoje „Procedural Modeling of Cities“ [10] šiam tikslui ir naudoja naudoja stochastinę, parametrinę L-Sistemą. Jų L-Sistemos objektai tai 3d objektų transformacijos operacijos, tokios kaip dydžio keitimas ar pajudinimas, tūrio ištraukimas (extrusion), išsišakojimas ir panaikinimas bei sudėtingesniems architektūriniams elementams, tokiems kaip stogai, langai ar antenos iš anksto paruošti modeliai.

Pav. 6 F--> F+F-F-FF+F+F-F grafinis vaizdas [11]

Tokios sistemos generacijos pavyzdys matomas 12 pav. Kaip matome, šis generacijos būdas taip pat leidžia nesunkiai optimizuoti gaunamus modelius – kiekvienas iteracijos žingsnis gali būti panaudojamas kaip vis detalesnio LOD pastato modelis.

Šios sistemos išeiga nusiunčiama į intepretatorių kuris simbolių eilutę paverčia į 3d objektus;

Pav. 7 Vienos iteracijos rezultatas [11]



Pav. 8 Pastatų generacija su L-sistema kur aksioma yra pastato maksimalus tūris, o kiekvienas paveiksliukas yra vienas papildomas iteracijos žingsnis [10]

### Greuter metodas

Stefanas Greuteris, Jeremy Parkeris, Nigelis Stewartas ir Geoffas Leachas publikavo tyrimą aprašantį kitokio pobūdžio nei L-sistemos pastatų generaciją.

Šiame metode pastatai kuriami pasinaudojant pseudo atsitiktinių skaičių generatoriumi – generatorius sugeneruoja „atsitiktinę“ skaičių eilutę panaudodamas iš vartotojo gautą raktą. Tas pats raktas visada sugeneruos tokius pačius skaičius. Skaičiai nusako įvairiais pastato charakteristikas: aukštį, aukštų skaičių ir t.t. [12]. Kadangi jų tyrime buvo kuriamas nesibaigiančio miesto generatorius, tai raktas buvo pastato koordinatė, kas leido visuomet toje pačioje vietoje sukurti/atkurti tokį patį pastatą.

Kiekvienas pastato aukštas sudaromas iš 2D grindų plano. Grindų planas tai ivairių formų poligonai kurių išvaizdą ir vietą nusako skaičių generatoriaus išeiga (grindų ploto generacija matoma 9 pav.). Atlikus nustatytą vieno aukšto grindų plano keitimo iteracijų kiekį, grindų plokštuma yra ištraukiama (extrude) paverčiant ją 3D tūriu, taip baigiant vieno aukšto generaciją [12]. Aukštai generuojami tol, kol pasiekiamas nustatytas pastato aukštis (aukštų generacija 10 pav.).

Kad geriau iliustruoti algoritmo veikimą, pateikiu pseudo kodą:

var pastatas;

ciklas(aukštų kiekis){

var grinduPlanas;

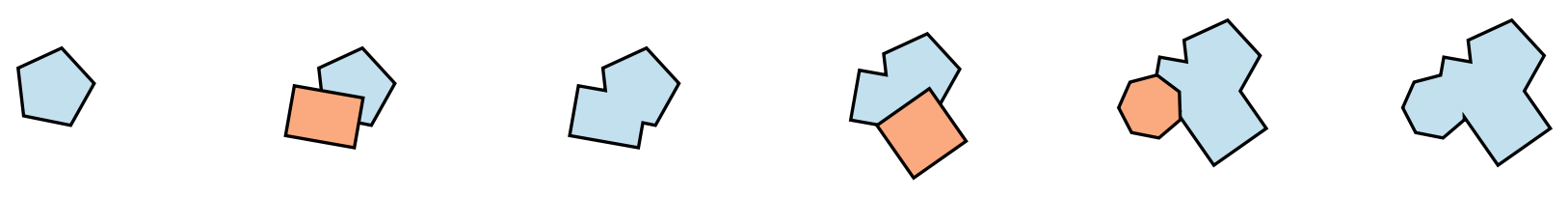
ciklas(iteracijų skaičius aukšte){

grinduPlanas = PridetiFigura(grinduPlanas);

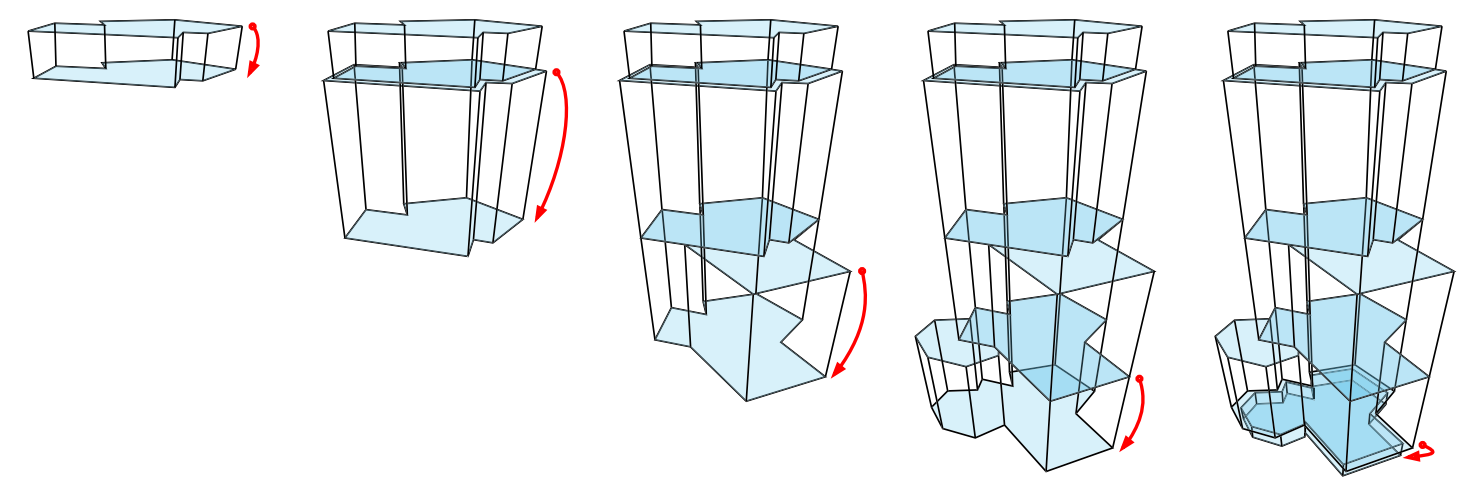
}

pastatas += IstrauktiTuri(grinduPlanas);

}



Pav. 9 Grindų plano generacija [12]

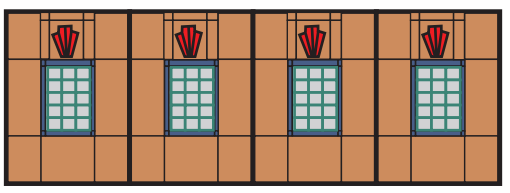
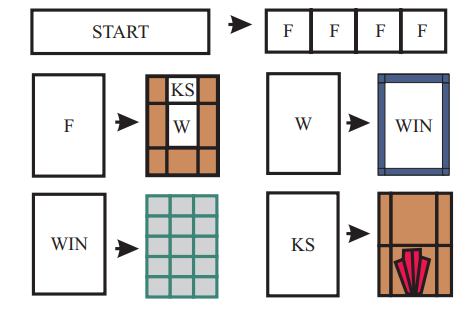


Pav. 10 Pastato aukštų generacija [12]

### Momentinė architektūra

Kaip ir L-sistema, tai dar viena formalia kalba (gramatika), aprašoma sistema, tačiau ji paremta ne simboliais, o geometrinėmis figūromis ir vadinama skaidymo gramatika. Šios gramatikos principas – figūra suskaidoma ir jos segementai pakeičiami kitomis figūromis kurios gali būti toliau skaidomos ir keičiamos kol pasiekiama kiekvienos figūros galutinė (nebeskaidoma) būsena. Sistema palaiko skirtingus architektūrinius stilius ir pastatų generaciją iš įvairiai sudeliotų stačiakampių, prizmių ir cilindrų. [13]

Pastatas paprastai sudeda is kelių pradinių figūrų, pavydžiui 2 aukštai po 4 stačiakampius. Tuomet pasitelkę 11 paveikslėlyje matomą antro aukšto generacijos vizualizaciją, matome kaip pradinis vieno stačiakampio plotas (F) skaidomas į tinklą kuriame išskiriami plotai langui (W) ir dekoracijai (KS). Tuomet W plotas skaidomas į rėmą ir patį lango stiklą, kas yra galutinės šių figūrų būsenos. KS parenkama stilių atitinkanti dekoracija ir taip pasiekiama šios dalis galutinė būsena. 12 paveikslėlis parodo gaunamą antro aukšto rezultatą.

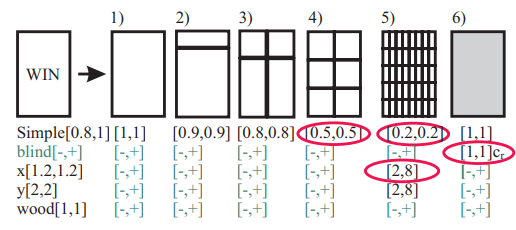


Pav. 11 Pradinio tūrio segmento skaidymo pavyzdys [13]

Pav. 12 Pradinio tūrio skaidymo pavyzdžio rezultatas [13]

Natūralu, kad generacija būtų sėkminga, algoritmas turi išlaikyti vieningą pastato stilių ir logiškai išdelioti architektūrinius elementus (duris, langus, stogą ir t.t.). Algoritmas turi sėkmingai atsirinkti taisyklęs iš visų tai būsenai pritaikomų taisyklių rinkinio. Tai pasiekiama prie gramatikos taisyklių prijungiant parametrus. Parametrai gali būti žemo lygio informacija – sienų spalva, arba aukšto lygio – pastato stilius. Aukšto lygio parametrai kuriant sistemą paprastai užpildomi ranka, žemo lygio apskaičiuojami automatiškai. Už parametrų perdavimą, tarp to pačio aukšto ar viso statinio segementų, atsakingos kontrolinės gramatikos taisyklės. Jų iškvietimas reguliuojamas specifinių parametrų prijungtų prie kiekvieno tūrio. Prieš kievieną tūrio skaidyma iškviečiamos kontrolinės taisyklės kurios padalina tėvinio tūrio parametrus visoms joms priklausančios erdvės (pvz. vienam aukštui) naujai suformuotoms figūroms (pvz. to aukšto langams).

Panagrinėkime konkretų objektų atrankos pavyzdį (13 pav.). Kaip matome yra išskirtas naujas plotas langui (WIN) su 5 parametrais (simple, blind, x, y, wood). Langams egzistuoja 6 skirtingi dizainai. „Simple“ parametras turi intervalą [0.8,1] kuris leidžia atmesti 4 ir 5 dizainus. Blind parametras su intervalu [−∞,∞], leidžia atmesti 6 dizainą. X pakartotinai atmeta 5 dizainą. Taigi lieka 3 lango variantai, iš kurių atsitiktinai galima pasiimti betkurį. Taigi tokia pastato segmentų išvaizdos atranka užtikrina tiek salyginai atsitiktinę pastato išvaizdą, tiek taisykles atitinkantį stilių.



Pav. 13 Segemento išvaizdos atmetimas pasinaudojant taisyklėmis (gramatika) [13]

## Skyriaus išvados

Apžvelgus dabartinę žaidimų rinkos situaciją, egzistuojančias procedūrinio pastatų generavimo programas bei algoritmus galima daryti keletą išvadų. Didelis mobilių žaidimų populiarumas reikalauja įrankių leidžiančių kurti juose naudojamus grafinius objektus. Natūralu, kad procedūrinė generacija tam tinka, kadangi veikia nepalyginti greičiau, bei gali pasiekti artimą ar net tokią pačią vizualinę kokybę kaip rankinis modeliavimas.

Procedūrinio generavimo tinkamumą žaidimų kūrimui rodo ir platus programų bei įrankių pasirinkimas įvairioms platformoms. Tiesa, visi šie įrankiai stengiasi būti kuo universalesni ir fokusuojasi į modernių, realistiškų pastatų generavimą. Atsižvelgus į minėta paprastų grafikų, low poly grafikų populiarumą, matosi, kad apžvelgti įrankiai neužpildo šio rinkos segmento. Taip pat, šie įrankiai naudotojo reikalauja išankstinio bazinių dalių rinkinio, kas su modeliavimu nesusipažinusiam gali tapti neįveikiama kliūtimi

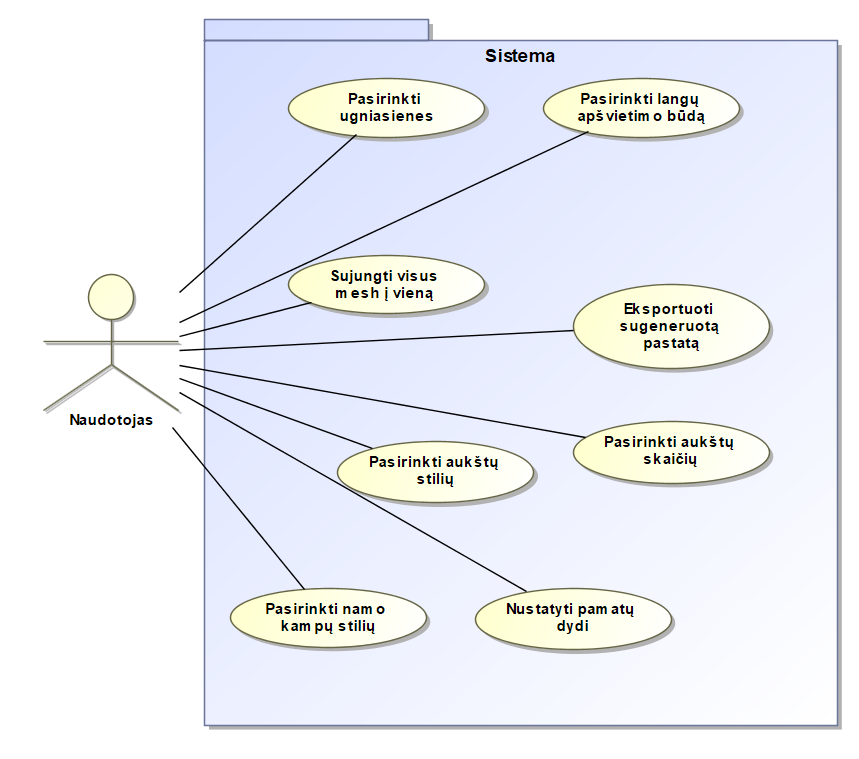
Taigi, šiame darbe pristatoma ir kuriama sistema bus orientuotą atlikti konkretų vaidmenį – generuoti low poly stilistikos viduramžių pastatus, nenaudodama jokių iš anksto sukurtų modelių rinkinio ir nereikalaudama rankinio tekstūravimo.

# Projektinė dalis

## Rekalavimų specifikacija

### Formuluojamos užduotys

#### Pagrindinės užduotys



#### Užduočių formulavimo kalbos reikalavimai

Vartotojo sąsaja yra Unity Editor plėtinys.

Vartotojo sąsaja suseda iš:

1. Nustatymų lango kuriame yra sudėtas visas vartotojo kontroliuojuomas funkcionalumas;
2. Unity scenos lango kuriame vaizduojamas generacijos rezultatas;
3. Unity konsolės kurioje rodomi pranešimai susiję su sistemos darbu;

Nustatymų lango sudėtis:

1. Mygtukai įjungti/išjungti ugniasienes
2. Mygtukas keisti langų apšvietimo išdėliojimą
3. Mygtukas įjungti/išjungti kampines kolonas
4. Mygtukas keisti aukštų stilių
5. Mygtukas įjungti naudotojo nustatyto pastato dydžio naudojimą
6. Slankiojantys intervalo pasirinkimai nustatyti pastato dydžiui
7. Minimalaus/maksimalaus aukštų skaičiaus nustatymo laukeliai
8. Mygtukas sujungti generuoto pastato meshus į vieną
9. Mygtukas pradėti generavimą
10. Mygtukas pradėti generavimo testą

#### Interfeiso darnos ir standartizavimo rekalavimai

Vartotojo sąsaja (nustatymų langas) turi tenkinti „MS Windows“ stadartus

#### Pranešimų formulavimo reikalavimai

Pranešimai sistemoje skirstomi į dvi kategorijas: informacinius ir klaidos. Pavyzdžiai:

* Informacinio: „Pastato generacija baigta. Trukmė 15ms“.
* Klaidos: „Pastato aukštų skaičius negali būti 0“

Pranešimai gali būti formuluojami anglų ir lietuvių kalbomis.

Kadangi įrankis yra Unity sistemos plėtinys, tai konsolė atvaizduoja tiek sistemoja aprašytus pranešimus tiek pačio Unity sukurtus.

#### Interfeiso individualizavimo reikalavimai

Ugniasienės –

Šviečiančių langų išdėliojimo stilius –

Kampinių kolonų būsena –

Aukštų dydžio stilius –

Norimas pastato dydis –

Aukštų skaičius -

### Funkciniai sistemos reikalavimai

#### Dalykiniai reikalavimai

#### Pagalbinės sistemos funkcijos

### Nefunkciniai reikalavimai

#### Vidinio interfeiso reikalavimai

##### Operacinės sistemos naudojimo reikalavimai:

Sistemai kurti turi būti naudojama Windows bei Android opracinės sistemos.

##### Sąveikos su duomenų bazėmis reikalavimai:

Duomenų bazė turi naudoti SQL kalbą ir turi būti talpina Microsoft Azure serveriuose

##### Dokumentų mainų reikalavimai:

Duomenų keitimas turi būti vygdomas json formatu.

##### Darbo kompiuterių tinkluose reikalavimai:

Programa turi naudoti TCP/IP protokalą veikimui internete

##### Sąveikos su kitomis programomis reikalavimai:

Sistema turi naudoti kitų sistemų meteorologinius duomenis bei išorinę duomenų bazę stovyklaviečių duomenims saugoti

##### Programavimo aplinkos reikalavimai:

Sistema kuriama Unity aplinkoje naudojant Visual Studio programavimo aplinką

#### Veikimo reikalavimai

##### Tikslumo reikalavimai:

Sistema turi būti kuriama taikantis į 1920 x 1080 skiriamosios gebos ekranus

Stovyklavietės pavadinimas negali viršyti 30 simbolių

Orų prognozės turi būti tikrinimos tik dvi savaites į priekį

##### Patikimumo reikalavimai:

Sistema turi pilnai veikti 95% laiko be trykių.

Sistema turi galėti naudotis 10000 naudotojų vienu metu

##### Robastiškumo reikalavimai:

Sistemos atkurimo laikas po ddos atakos turi būti – iki dviejų valandų.

Trykius iššaukiančių įvykių procentas turi būti 1%.

Tikimybė, kad trykio metu bus sugadinti duomenys – 1%.

##### Našumo reikalavimai.

##### Reakcijos laikas:

Vidutinis reakcijos laikas į vartotojo sąsajos paspaudimą turi būti 5ms.

Lango pakitimo animacijos laikas turi neviršyti 500ms.

##### Pralaidumas (throughput):

Sistema vienu metu turi galėti apdoroti 10000 vartotojų paieškos užklausų.

##### Masto keitimas (scalability):

Kadangi sistema veiks ant Azure serverio, resursai turi atsirasti dinamiškai.

#### Diegimo reikalavimai

##### Instaliuojamumas

Sistema turi būti instaliuojama maksimaliai per minutę naudojantis android arba apple programėlių parduotuve.

##### Įsisavinamumas:

Vartotojas, kuris moka naudotis bazinėmis telefono funkcijomis, turi galėti įsisavinti visą programą ne ilgiau nei per 5 minutes.

##### Išmokstamumas

Sistema turi turėti ne daugiau nei penkis langus, todėl funkcionalumas bus isisąvintas per vieną dieną.

#### Ruošinio reikalavimai

##### Programėlės pasiekimas:

Programėlė turi būti pasiekiama tiek „Android“ bei „Apple“ telefonų operacinių sistemų oficialiose parduotuvėse.

#### Aptarnavimo ir priežiūros reikalavimai

Sistema turės aptarnauti sistemos sąvininkas, tais atvejais kuomet reikės ištrinti kokybės standartų neatitinkančias stovyklavietes

##### Taisomumo

Trikio ištaisymas turėtų užimti ne ilgiau nei 5min.

##### Keičiamumo

Kadangi sistema labai paprasta, turėtų užimti ne ilgiau nei viena diena

##### Plečiamumo

Kadangi sistema yra debesų sistemoje, ji dinamiškai turės prisitaikyti prie padidėjusio vartotjo kiekio iki 100 milijonų.

##### Perkeliamumo

Perkelti duomenų bazę į kitą sistemą turi užimti vieną dieną.

##### Testuojamumo

Testams duomenų bazei sukurti turi užtrukti vieną savaitę.

Testams aplikacijos „front-endui“ sukurti turi užtrukti vieną savaitę.

Testams „back-endui“ sukurti turi užtrukti vieną savaitę.

#### Tiražuojamumo reikalavimai

Turi būti idiegta nemokama „MySQL“ duomenų bazės valdymo įranga.

#### Apsaugos reikalavimai

Sistemos duomenų baze turi galėti naudotis tik vienas autorizuotas vartotojas – sistemos sąvininkas.

#### Juridiniai reikalavimai

L.R. asmens duomenų teisinės apsaugos įstatymas: 6. Straipsnis Aplikacijoje nebus saugomi nei vartotojų, nei darbuotojų asmeniniai duomenys todėl jokias atvejais nebus pažeidžiami individo duomenų apsaugos įstatymai ar kiti LR teisės aktai.

## Programų sistemos projektas

### Programų sistemos projektiniai reikalavimai

#### Programų sistemos dekompozicija

#### Reikalavimų lokalizavimo matrica

#### Reikalavimų ryšio matrica

### Programų sistemos architektūra

#### Užduotys ir jų vykdymo scenarijai

#### PS struktūros modelis

### Programų sistemos maketai

~Tikslas:



<https://assetstore.unity.com/detail/3d/props/exterior/low-poly-medieval-house-123281>

# Testavimas

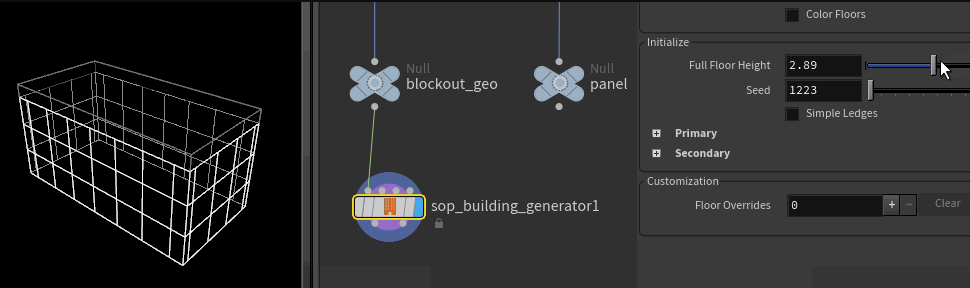
# Išvados ir siūlymai

# Literatūra ir šaltiniai

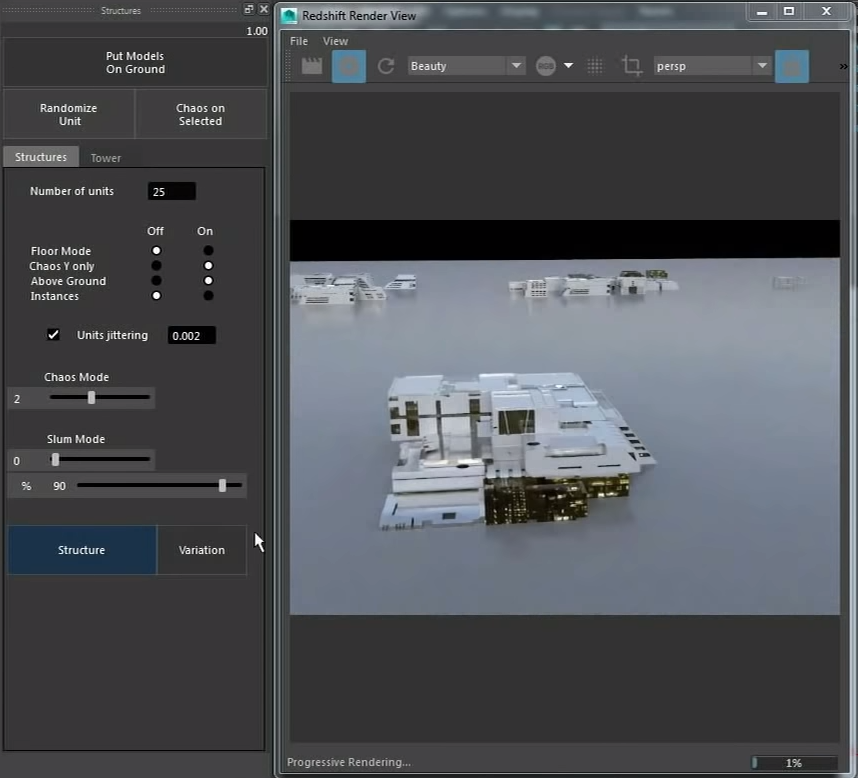
|  |  |
| --- | --- |
| [1] | „newzoo,“ [Tinkle]. Available: https://newzoo.com/insights/articles/the-global-games-market-will-generate-152-1-billion-in-2019-as-the-u-s-overtakes-china-as-the-biggest-market/. [Kreiptasi 27 12 2019]. |
| [2] | J. Couture, „Gamasutra,“ 01 07 2016. [Tinkle]. Available: https://www.gamasutra.com/view/news/273904/Why\_are\_so\_many\_devs\_employing\_a\_retro\_lowpoly\_mid1990s\_aesthetic.php. |
| [3] | E. Narcisse, „Kotaku,“ [Tinkle]. Available: https://kotaku.com/why-new-video-games-still-cost-60-1545590499. |
| [4] | [Tinkle]. Available: https://panoramastreetline.com/news/deutsche-fachwerkstadt-fachwerkstrasse-panorama. |
| [5] | „SideFX,“ pastatų generatorius, [Tinkle]. Available: https://www.sidefx.com/tutorials/building-generator/. [Kreiptasi 20 11 2019]. |
| [6] | „Maya Scructures,“ pastatų generatorius, [Tinkle]. Available: https://gumroad.com/l/UELQtt. [Kreiptasi 20 11 2019]. |
| [7] | „Building Generator,“ pastatų generatorius, [Tinkle]. Available: http://tysonibele.com/Main/BuildingGenerator/buildingGen.htm. [Kreiptasi 20 11 2019]. |
| [8] | „BuildR,“ pastatų generatorius, [Tinkle]. Available: http://support.jasperstocker.com/buildr2/. [Kreiptasi 20 11 2019]. |
| [9] | „SceneCity,“ miestų generatorius, [Tinkle]. Available: https://www.cgchan.com/store/scenecity. [Kreiptasi 20 11 2019]. |
| [10] | P. M. Yoav I H Parish, „Procedural Modeling of Cities,“ [Tinkle]. Available: https://cgl.ethz.ch/Downloads/Publications/Papers/2001/p\_Par01.pdf. [Kreiptasi 21 11 2019]. |
| [11] | P. Bourke, „L-System User Notes,“ [Tinkle]. Available: http://paulbourke.net/fractals/lsys/. [Kreiptasi 21 11 2019]. |
| [12] | J. P. N. S. G. L. Stefan Greuter, „Real-time Procedural Generation of ‘Pseudo Infinite’ Cities,“ [Tinkle]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.88.7296&rep=rep1&type=pdf. [Kreiptasi 21 11 2019]. |
| [13] | M. W. F. X. S. W. R. Peter Wonka, „Instant Architecture,“ [Tinkle]. Available: https://hal.inria.fr/inria-00527500/file/instant\_architecture.pdf. [Kreiptasi 22 11 2019]. |

**PRIEDAI**

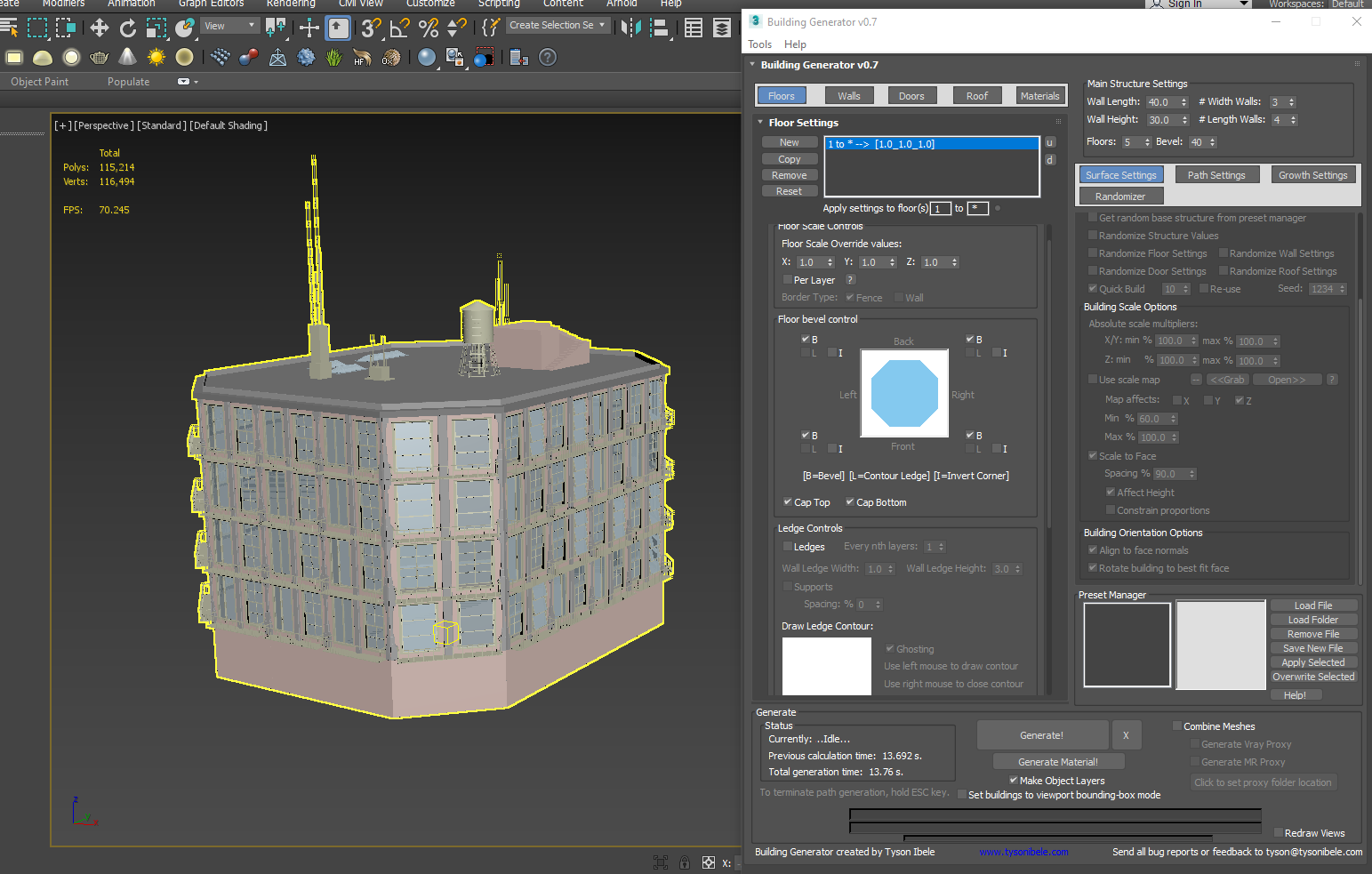
**1 PRIEDAS. Pastatų ir miestų generatorių priedai**



Priedas 1 Houdini building generator varotojo sąsajos pavyzdys [5]



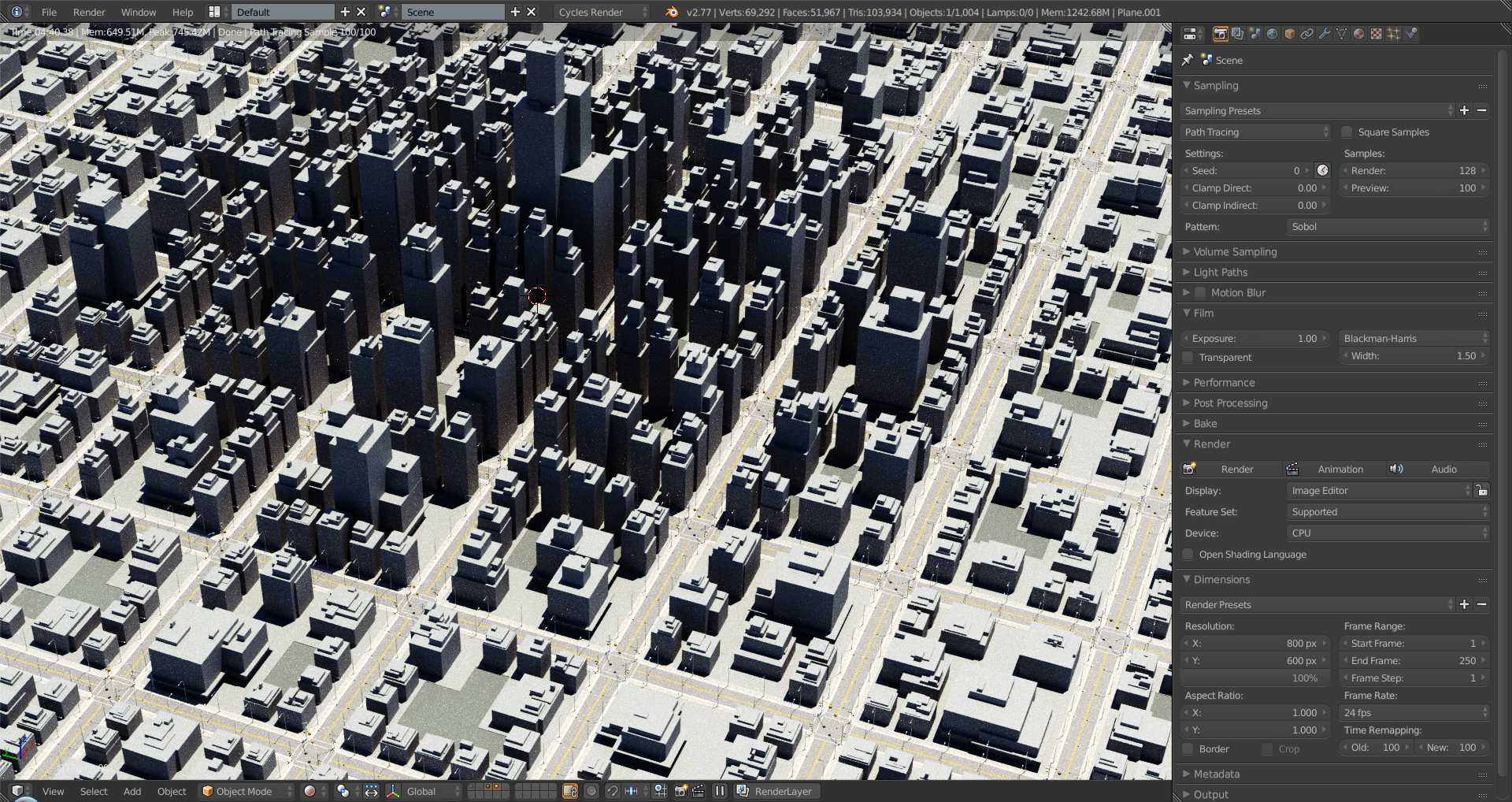
Priedas 2 Maya structures vartotojo sąsaja ir generacijos pavyzdys [6]



Priedas 3 Building Generator v0.7 vartotojo sąsaja ir generacijos pavyzdys

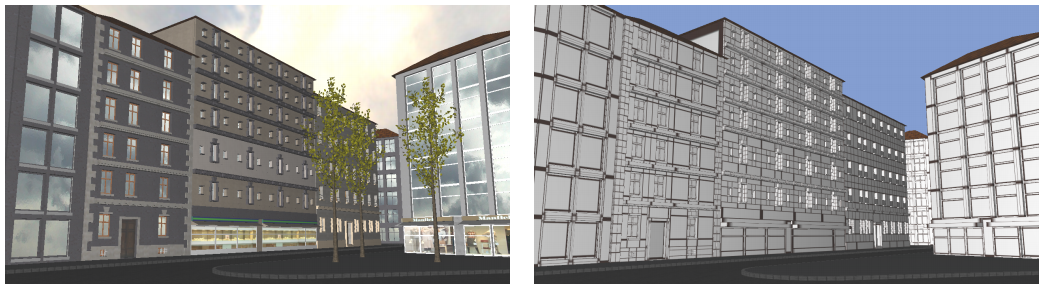


Priedas 4 SceneCity generacijos pavyzdys [9]



Priedas 5 SceneCity procedūriškai sugeneruoti pastatai [9]

**2 PRIEDAS. Generacijos algoritmų priedai**



Priedas 6 "Momentinės architektūros" algoritmo generacijos pavyzdys [13]