VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA STROJNÍHO INŽENÝRSTVÍ
ÚSTAV MATEMATIKY

FACULTY OF MECHANICAL ENGINEERING
INSTITUTE OF MATHEMATICS

MATEMATICKÉ METODY MODELOVÁNÍ MORFOLOGIE JEHLIČNANŮ

MATHEMATICAL METHODS OF MORPHOLOGY MODELLING OF CONIFEROUS TREES

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. RŮŽENA JANOUTOVÁ

VEDOUCÍ PRÁCE
SUPERVISOR

doc. PaedDr. DALIBOR MARTIŠEK, Ph.D.

BRNO 2012

Vysoké učení technické v Brně, Fakulta strojního inženýrství

Ústav matematiky
Akademický rok: 2011/2012

ZADÁNÍ DIPLOMOVÉ PRÁCE

student(ka): Bc. Růžena Janoutová

který/která studuje v **magisterském navazujícím studijním programu**

obor: **Matematické inženýrství (3901T021)**

Ředitel ústavu Vám v souladu se zákonem č.111/1998 o vysokých školách a se Studijním a zkušebním řádem VUT v Brně určuje následující téma diplomové práce:

Matematické metody modelování morfologie jehličnanů

v anglickém jazyce:

Mathematical methods of morphology modelling of coniferous trees

Stručná charakteristika problematiky úkolu:

Práce naváže na výsledky dosažené v práci (3). Výsledky bakalářské práce budou použity speciálně k simulování morfologie jehličnatých stromů, především smrků. Předpokládá se spolupráce s Centrem výzkumu globálních změn AV ČR v Brně.

Cíle diplomové práce:

Cílem práce je vytvořit vizualizační algoritmy vhodné pro studium morfologie jehličnanů, a to pomocí L-systémů. Budou vytvořeny podpůrné vizualizační nástroje pro vybrané třídy L-systémů a jejich implementace.

Seznam odborné literatury:

1. Žára, J. a kol.: Moderní počítačová grafika, Computer Press 1998
2. Kolka, M.: L-systems: new results and application, BUT Brno, 2004
3. Janoutová, R.: Matematické modelování pomocí L-systémů, VUT v Brně 2010, bakalářská práce

Vedoucí diplomové práce: doc. PaedDr. Dalibor Martišek, Ph.D.

Termín odevzdání diplomové práce je stanoven časovým plánem akademického roku 2011/2012.

V Brně, dne 19.11.2010

L.S.

prof. RNDr. Josef Šlapal, CSc.	prof. RNDr. Miroslav Doupovec, CSc., dr. h. c.
Ředitel ústavu	Děkan fakulty

ABSTRAKT

Cílem práce byla tvorba jehličnatého stromu nedestruktivní metodou umožňující popis struktury dospělého smrkového porostu. Z poskytnutých dat byl po jejich zpracování vytvořen model L-systému, kterým se vytvářely větve stromu. V algoritmu Python skriptu byly následně vygenerovány parametry nutné k vytvoření modelu stromu v grafickém softwaru Blender. Model jehličnatého stromu se podařilo úspěšně vygenerovat. Jeho paměťová náročnost je velká, ale pro účely vytvoření modelu to není zásadní problém.

KLÍČOVÁ SLOVA

L-systémy, formální jazyky, jehličnaté stromy, aplikace L-systémů, Blender, Python, zpracování dat, vizualizace dat

ABSTRACT

The thesis was focused on creation of a coniferous tree by nondestructive method allowing description of structure of adult spruce trees. After processing provided data we created a model of L-system which creates a tree branch. Thereafter, a Python script generated parameters which were required for the creation of the model of the tree in graphical software Blender. Model of coniferous tree was sucesfully generated. Its memory requirements are high but for our purposes this is not an essential problem.

KEYWORDS

L-systems, formal languages, coniferous trees, L-system application, Blender, Python, data processing, data visualization

JANOUTOVÁ, R. *Matematické metody modelování morfologie jehličnanů*. Brno: Vysoké učení technické v Brně, Fakulta strojního inženýrství, 2012. 51 s. Vedoucí bakalářské práce doc. PaedDr. Dalibor Martišek, Ph.D.

PROHLÁŠENÍ

Prohlašuji, že svou diplomovou práci na téma „Matematické metody modelování morfologie jehličnanů“ jsem vypracovala samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušila autorská práva třetích osob, zejména jsem nezasáhla nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědoma následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.

Brno ....

(podpis autora)

Děkuji svému školiteli, doc. PaedDr. Daliboru Martiškovi, Ph.D., za odborné konzultace a vedení mé diplomové práce.

OBSAH

ÚVOD

Principem L-systémů se jako první zabýval maďarský biolog Aristid Lindenmayer. Vycházel z myšlenky, že v oplodněném vajíčku se nachází program, který řídí vývoj organizmu [1]. L-systémy pracují na principu formálních gramatik. V grafické interpretaci se dají chápat jako tzv. želví grafika, kde jednotlivé symboly popisují úkony pro želvu. Jsou často využívány na modelování různých přírodních jevů, filmových efektů a mnoho dalšího. Škálou využití se zabývala moje bakalářská práce [1]. Tato práce se zabývá využitím L-systémů v praxi. Ve spolupráci s Centrem výzkumu globální změny AV ČR, v.v.i. (veřejná výzkumná instituce; dále označováno jako CVGZ), byl vytvořen matematický model smrku. Tento model byl realizován v programovacím jazyku Python pro implementaci do grafického programu Blender, který splňuje požadavky pro formát výstupu modelu.

V práci jsou nejdříve definovány formální jazyky, gramatiky a následně i L-systémy. L-systémy mají různé charakteristiky, a proto byly definovány alespoň ty nejpoužívanější, především ty, se kterými se pracovalo v modelu. Práce popisuje problém, pro který se model vytvářel, zpracování zadaných dat na data použitelná pro model a hlavně implementaci do L-systémů. Je popsána charakteristika jednotlivých znaků abecedy a tvorba přepisovacích pravidel. Je uvedena volba a nároky na výběr softwaru a programovacího jazyka, v němž je sestaven algoritmus na tvorbu modelu a jeho popis.

Cílem práce bylo vytvořit vizualizační algoritmy vhodné pro studium morfologie jehličnanů, a to pomocí L-systémů. Dále byly vytvořeny podpůrné vizualizační nástroje pro vybrané třídy L-systémů a jejich implementace.

1 TEORETICKÁ ČÁST

1.1 Formální jazyky a gramatiky

L-systémy jsou založeny na teorii Formálních jazyků. Proto budou nejdříve definovány pojmy související s Formálními jazyky, aby je pak bylo možno využít při zavedení gramatik a L-systémů. Základní pojmy lineární algebry a diskrétní matematiky jsou považovány za známé. Jejich definice se proto nebudou uvádět a bude se definovat až abeceda.

Definice 1.1. Libovolnou konečnou množinu Σ nazveme ***abecedou***. Prvky abecedy Σ nazýváme ***symboly***.

Příkladem abecedy je třeba množina $\{a, b\}$, množina číslic $\{0, 1, \ldots, 9\}$ nebo prázdná množina $\emptyset$.

Definice 1.2. Konečnou posloupnost prvků množiny Σ nazveme ***slovo*** nad abecedou Σ.

Například $aabb$ je slovo nad abecedou $\{a, b\}$.

Definice 1.3. Množinu všech neprázdných slov označíme Σ^+. Prázdnou posloupnost ε nazveme ***prázdné slovo***. Množinu všech slov nad abecedou Σ označíme $\Sigma^* = \Sigma^+ \cup \{ \varepsilon \}$.

Platí např.

$$\begin{aligned} \{a\}^* &= \{\varepsilon, a, aa, aaa, aaaa, \ldots\} \\ \{a\}^+ &= \{a\}^* \setminus \{\varepsilon\} \\ \{0,1\}^* &= \{\varepsilon, 0, 1, 00, 01, 10, 11, 000, 001, 010, 011, \ldots\} \end{aligned}$$

Definitoricky dále klademe $\emptyset^* = \{\varepsilon\}$ a $\emptyset^+ = \emptyset$.

Definice 1.4. Počet symbolů ve slově α nazveme ***délkou slova*** a označíme $|\alpha|$. Definujeme $|\varepsilon| = 0$.

Například slovo $abaaba$ má délku $|abaaba| = 6$.

Definice 1.5. Počet výskytů symbolu a ve slově α definujeme jako ***četnost symbolu*** a značíme ji $\#_a(\alpha)$.

Například $\#_b(abaaba) = 2$.

Definice 1.6. Nechť Σ je abeceda a nechť $\alpha = a_1 a_2 \ldots a_n$ a $\beta = b_1 b_2 \ldots b_m$ jsou slova nad abecedou Σ, kde $m, n \in \mathbb{N}, a_i, b_j \in \Sigma$. Slovo $\gamma = a_1 a_2 \ldots a_n b_1 b_2 \ldots b_m$ nazýváme ***zřetězení*** slov α, β. Operaci zřetězení značíme $\cdot$ a je dána předpisem $\alpha \cdot \beta = \alpha\beta$.

Například zřetězením slov *abba* a *bba* obdržíme slovo *abbabba*. Operace zřetězení je asociativní, tj. $u \cdot (v \cdot w) = (u \cdot v) \cdot w$ pro libovolná slova u, v, w. Dále se ε chová jako neutrální prvek, tj. $u \cdot \varepsilon = \varepsilon \cdot u = u$ pro libovolné u.

Věta 1.1. $(\Sigma^*, \cdot)$ je grupoid s neutrálním prvkem ε.

Věta 1.2. Každé neprázdné slovo nad abecedou Σ lze získat zřetězením nenulového počtu symbolů této abecedy.

Definice 1.7. Nechť α, β jsou slova nad abecedou Σ, řekneme, že α je ***podslovo*** slova β, jestliže existuje $lc, rc \in \Sigma^*$ tak, že:

$$\beta = lc\alpha rc$$

Pokud navíc $lc = \varepsilon$, říkáme, že slovo α je ***předponou*** slova β. Je-li $rc = \varepsilon$ nazveme α ***příponou*** slova β.

Například *aa* je předponou *aabab*, zatímco 11 není ani podslovem slova 01010.

Definice 1.8. Libovolnou podmnožinu L množiny Σ^* nazýváme ***jazykem*** nad abecedou Σ.

Například $\{10, 1, 011101\}$ je jazyk nad abecedou $\{0, 1\}$. Jazyky mohou ovšem být i nekonečné, např. $\left\{\alpha \in \{a, b\}^* \,\middle|\, \#_a(\alpha) = \#_b(\alpha)\right\}$ je jazyk nad abecedou $\{a, b\}$ obsahující všechna slova, ve kterých se a i b vyskytují ve stejném počtu, tedy např. $aababb, \varepsilon$ jsou prvky tohoto jazyka, zatímco $a, abbaa$ nikoli. Prázdná množina je jazyk nad libovolnou abecedou.

Definice 1.9. Nechť Σ je abeceda a máme $lc, rc \in \Sigma^*$. Pak uspořádanou dvojici $(lc, rc) \in \Sigma^* \times \Sigma^*$ nazýváme ***kontext*** nad abecedou Σ, jestliže:

$$lc\alpha rc \in \Sigma^*,$$

kde $\alpha \in \Sigma^*$. Řekneme, že ***kontext*** (lc, rc) ***přijímá slovo*** α ***v jazyce*** L, jestliže navíc platí:

$$lc\alpha rc \in L,$$

kde L je jazyk nad abecedou Σ. lc nazýváme ***levý kontext*** a rc ***pravý kontext***.

Definice 1.10. ***Gramatika*** $\mathcal{G}$ je čtveřice (N, T, P, σ), kde N je neprázdná konečná množina ***neterminálních symbolů***. T je konečná množina ***terminálních symbolů*** taková, že $N \cap T = \emptyset$. Sjednocením N a T obdržíme množinu všech symbolů gramatiky, kterou označujeme Σ. $P \subseteq \Sigma^* \cdot N \cdot \Sigma^* \times \Sigma^*$ je konečná množina ***přepisovacích pravidel***. Pravidlo (α, β) obvykle zapisujeme ve tvaru $\alpha \to \beta$. $\sigma \in N$ je speciální ***počáteční*** neterminál.

Pozn.: *Význam terminálních a neterminálních symbolů bude uvedena později.*

Pozn.: *Operaci zřetězení na množinách symbolů N a T chápeme jako $N \cdot T = \{u \cdot v | u \in N \wedge v \in T\}$.*

1.2 L-systémy

Po zavedení formální jazyků a gramatik bude uvedena definice L-systému. Pro lepší interpretaci se některé výrazy označují jinak než v terminologii formálních jazyků a gramatik. Definice, příklady a obrázky jsou převzaty z [1]. Další informace lze nalézt i v [3].

Definice 1.11. ***L-systémem*** rozumíme uspořádanou trojici $\mathcal{L}= (\Sigma, P, \sigma)$, kde Σ značí konečnou množinu symbolů, kterou nazýváme *abecedou*, P je množina přepisovacích pravidel tvaru $p : \alpha \to \beta$; $\alpha \in \Sigma$; $\beta \in \Sigma^*$; $p \in P$ a $\sigma \in \Sigma^+$ je axiom (počáteční neterminál).

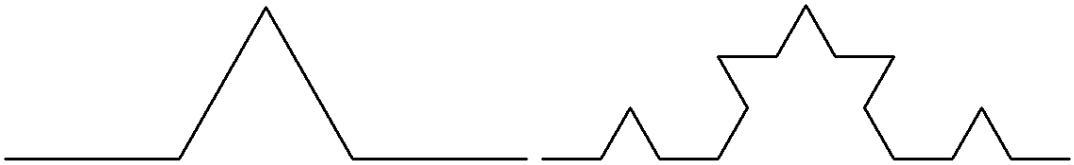
Pozn.: *σ můžeme chápat jako nultou iteraci systému. Prvek ε, můžeme v systému využít pro mazání symbolů. Když přepisovací pravidlo bude mít tvar $p : \alpha \to \varepsilon$, pak symbol náležící tomuto pravidlu bude smazán (nahrazen prázdnou posloupností). Jednou iterací L-systému chápeme přepis všech neterminálních znaků v řetězci.*

Následující příklad interpretuje L-systém jako grafický nástroj. Symboly můžeme chápat jako příkazy pro želvu. Uvedeme několik základních příkazů, krok vpřed a natočení. Krok vpřed značíme písmenem F a natočení o daný úhel do kladného směru $+$ a do záporného směru $-$.

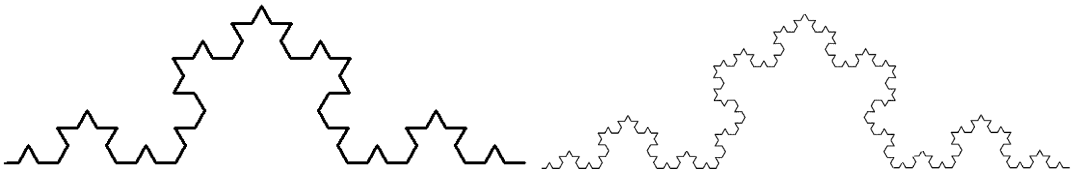
Příklad 1.1. Zvolíme množinu Σ, která obsahuje prvky $\{F, +, -\}$, kde F odpovídá v grafickém výstupu posunutí dopředu s vykreslováním stopy, $+$ je chápáno jako otočení o úhel 60° v kladném směru (po směru hodinových ručiček) a $-$ do záporného směru. P obsahuje pouze jedno pravidlo, které je definováno předpisem $p : F \to F - F + +F - F$, což znamená, že v každé iteraci znak F nahradíme řetězcem $F - F + +F - F$. σ je zadána znakem F. Zadání tedy vypadá takto:

$$\text{L-systém: } \mathcal{L} = (\Sigma, P, \sigma);$$
$$\Sigma = \{F, +, -\};$$
$$P = p, \text{ kde } p : F \to F - F + +F - F;$$
$$\sigma = F;$$
$$\text{úhel} = 60°;$$

Tímto L-systémem vykreslenou křivku nazýváme Kochova křivka podle švédského matematika Nielse Fabiana Helge von Kocha. Na obr. 1.1 a 1.2 vidíme první čtyři iterace, vygenerované v programu, který je součástí práce [1].



Obr. 1.1: Kochova křivka: 1. a 2. iterace



Obr. 1.2: Kochova křivka: 3. a 4. iterace

Rozšíření výše popsaných L-systémů umožňuje jejich větvení. Želvě přidáme zásobník, kam si bude ukládat souřadnice pozic a úhlů na těchto místech, na které se bude vracet. Ve chvíli, kdy želva narazí na [, se do zásobníku na poslední místo zapíše pozice a úhel a ve chvíli, kdy narazí na] , se vrátí na pozici a načte úhel, jež jsou uloženy na posledním místě v zásobníku.

1.3 Klasifikace L-systémů

Klasifikace L-systémů není v literatuře jednotná. Některé typy jsou však přijímány obecně. Uvedeme si některé z nich, které budeme následně používat. Definice a příklady jsou převzaty z bakalářské práce [1], kde jsou uvedeny i další charakteristiky.

Definice 1.12. Uvažujme přepisovací pravidlo $p \in P$ tvaru $p : lc < \alpha > rc \to \beta$, kde lc je levý kontext a rc pravý kontext, $lc, rc \in \Sigma^*$, v němž symbol α bude přepsán pouze tehdy, když stojí uprostřed kontextu, L-systém s těmito pravidly nazýváme ***kontextový***. Značíme ho ***IL-systém***. L-systém, který není kontextový, nazýváme bezkontextový a značíme ho ***0L-systém***.

Příklad 1.2. V tomto příkladu ukážeme využití IL-systému k přenášení signálu (symbolu) v řetězci:

$$\text{L-systém: } \mathcal{L} = (\Sigma, P, \sigma);$$
$$\Sigma = \{a, b\};$$
$$P = \{p_1, p_2\}, \text{ kde } p_1 : b < a \to b;\ p_2 : b \to a;$$
$$\sigma = baaaaa;$$

Vypíšeme několik iterací včetně σ:

$$baaaaa$$
$$abaaaa$$
$$aabaaa$$
$$aaabaa$$
$$aaaaba$$

Definice 1.13. V přepisovacím pravidle $p \in P$ ve tvaru $p : \alpha \to \beta$ nazveme α ***levou stranou*** přepisovacího pravidla a β nazveme ***pravou stranou*** přepisovacího pravidla. L-systém neobsahující v množině přepisovacích pravidel P žádná dvě pravidla se stejnou levou stranou nazýváme ***deterministický***. Značíme jej ***DL-systém***.

Definice 1.14. Symbol nazveme ***terminální***, pokud se v dalších iteracích nepřepisuje, tzn. neexistuje přepisovací pravidlo s levou stranou rovnou tomuto symbolu. Symbol nazveme ***neterminální***, pokud se v dalších iteracích přepisuje, tzn. existuje přepisovací pravidlo s levou stranou rovnou tomuto symbolu. Rozšířením abecedy Σ o množinu T terminálních symbolů dostaneme ***rozšířený*** L-systém a značíme jej ***EL-systém***. Množinu neterminálních symbolů označíme N. Rozšířeným L-systémem rozumíme uspořádanou čtveřici $\mathcal{L} = (N, T, P, \sigma)$.

Pozn.: *Platí* $T \cap N = \emptyset \wedge T \cup N = \Sigma^*$.

Definice 1.15. ***Parametrickým*** L-systémem rozumíme uspořádanou čtveřici $\mathcal{L} = (\Sigma, P, \sigma, X)$, kde $X \subseteq \mathbb{R}$.

Pozn.: *Pravidla* $p \in P$ *pak mají tvar* $p : \alpha(x_1, x_2, \ldots x_n) : b(x_1, x_2, \ldots, x_n) \to \beta(x_1, x_2, \ldots x_n)$, *kde* $m, n \in \mathbb{N}$ *a* b *nabývá hodnot* $\{0, 1\}$ *a reprezentuje podmínku na parametry.*

Definice 1.16. L-systém obsahující v množině přepisovacích pravidel P více pravidel se stejnou levou stranou nazýváme ***nedeterministický***. Přepisovací pravidlo p má tvar

$$p_{id} : \alpha \to \beta : \mu_{id},$$

kde

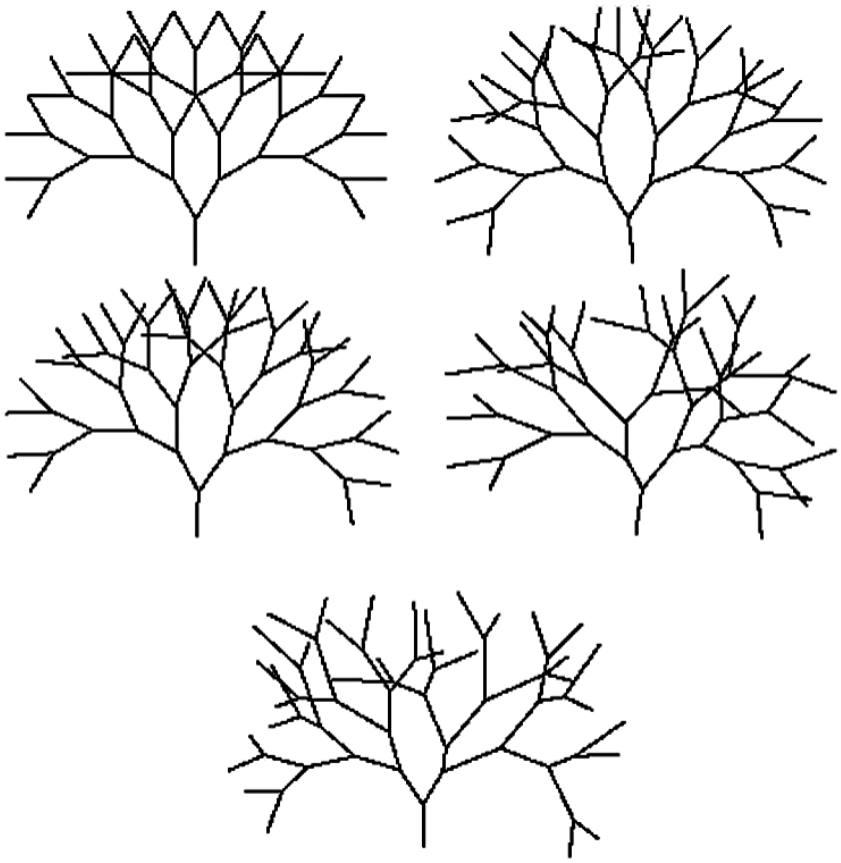
$$id \in \mathbb{N}, \mu_{id} \in \langle 0, 1 \rangle.$$

Platí

$$\sum_{id=1}^{n} \mu_{id} = 1,$$

kde $n \in \mathbb{N}$ reprezentuje počet přepisovacích pravidel p se stejnou levou stranou, id je index pravidla p a μ_{id} je pravděpodobnost s jakou se přepíše pravidlo p s indexem id.

Při grafické interpretaci L-systémů a významu terminálních znaků + a − otočení želvy, zavedeme ***náhodné otočení*** následovně: náhodné otočení znamená, že se úhlu natočení přiřadí náhodné rozdělení s danými parametry. Náhodné rozdělení se používá nejčastěji rovnoměrné nebo normální, výběr náhodného rozdělení závisí na konkrétním modelu. Důsledky náhodného natočení jsou znatelné na obr. 1.3.



Obr. 1.3: Srovnání deterministického (vlevo nahoře) se stochastickým L-systémem - použití náhodného otočení, převzato z [1]

Jelikož jsme přepisovací pravidlo značně obohatili, uvedeme příklad, jak by mohla vypadat některá přepisovací pravidla a první iterace, jež jich využije.

Příklad 1.3. L-systém: $\mathcal{L} = (\Sigma, P, \sigma, X)$;

$$X = \{x\};$$
$$\Sigma = \{\alpha(x), \beta(x)\};$$
$$\sigma : \alpha(1)\beta(3)\alpha(5);$$
$$P = \{p_1, p_2, q\};$$
$$p_1 : \alpha(x) \to \alpha(x+1) : 0,4;$$
$$p_2 : \alpha(x) \to \beta(x-1) : 0,6;$$
$$q : \alpha(x_1) < \beta(x_2) > \alpha(x_3) : x_2 < 4 \to \beta(x_1 + x_3)\,[\alpha(x_2)]\,;$$

Pravidla p_1 a p_2 vnášejí do přepisování náhodnost. Tato dvě pravidla symbol $\alpha(x)$ přepíší s pravděpodobností $0,4$ na symbol $\alpha(x+1)$ a s pravděpodobností $0,6$ na symbol $\beta(x-1)$. Třetí pravidlo přepíše symbol $\beta(x_2)$ na posloupnost symbolů $\beta(x_1+x_3)$ $[\alpha(x_2)]$ za daných kontextových podmínek tj., když symbol $\beta(x_2)$ bude mít levý kontext symbol $\alpha(x_1)$ a pravý kontext symbol $\alpha(x_3)$, a také pokud bude splněna podmínka $x_2 < 4$.
První iterace pak může vypadat například takto:

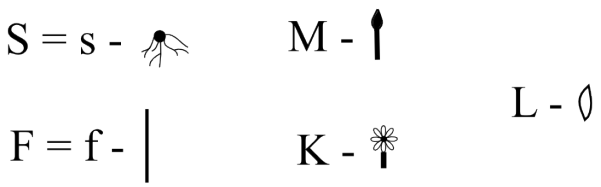
$$\alpha(1)\beta(3)\alpha(5) \Rightarrow \alpha(2)\beta(6)[\alpha(3)]\alpha(4).$$

Předchozí příklad byl názorný pro použití přepisovacích pravidel, ale pro reálné využití si uvedeme grafický příklad, jak může taková aplikace zjednodušeně vypadat. Grafický příklad můžeme chápat jako růst rostliny. Rostlina zde roste od semínka až po kvetoucí rostlinu. V následujícím příkladu je vidět využití rozšířených, kontextových a parametrických L-systémů. Stochastické L-systémy jsou tentokrát vynechány, ale příklad by se dal lehce modifikovat na stochastický L-systém. Příklad i obrázky v něm byly vytvořeny v rámci diplomové práce.

Příklad 1.4. L-systém: $\mathcal{L} = (\Sigma, P, \sigma, X)$,

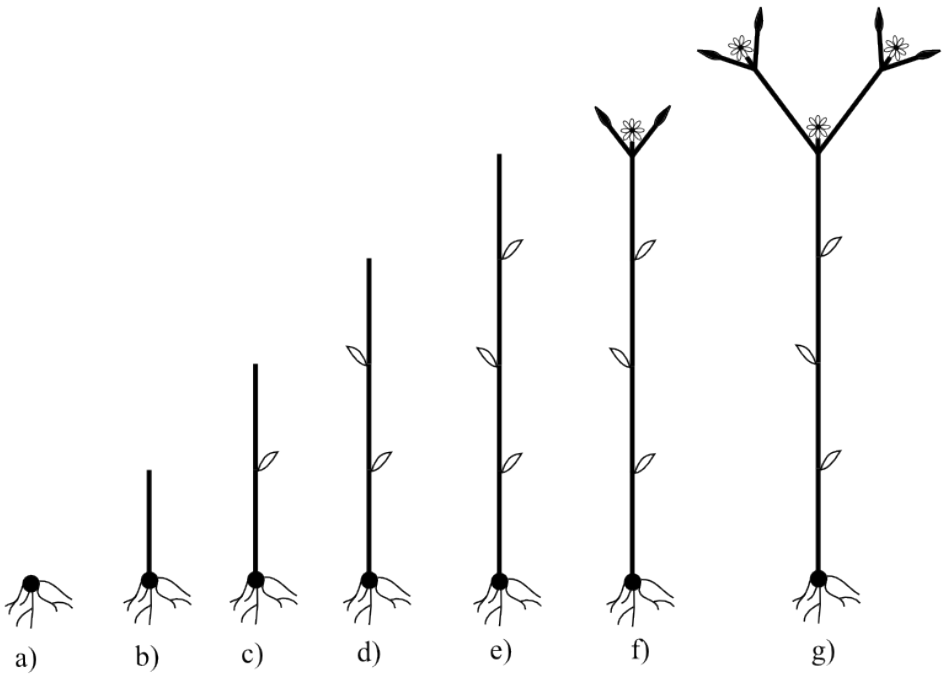
$$X = \{v, d\},$$
$\Sigma = \{s, S, f(v,d), F(v,d), L, M, K, [,], +, -\}$, kde $N = \{s, f(v,d), M\}$ a
$$T = \{S, F(v,d), L, K, [,], +, -\}$$
$$\sigma : s,$$
$$P = \{p, q, r, t, u\},$$
$$p : s \to Sf(1,1),$$
$$q : -L] < f(v,d) : v < 4 \to F(v,d)[+L]f(v+1,d),$$
$$r : +L] < f(v,d) : v < 4 \to F(v,d)[-L]f(v+1,d),$$
$$t : f(v,d_1) : v \geq 4 \to F(v,d_1)[-M][+M]K,$$
$$u : F(v,d_1)[\pm < M :\to F(v, \tfrac{2}{3}d_1)[-M][+M]K,$$

Grafický význam jednotlivých symbolů je ilustrován na obr. 1.4. Jak je již na obrázku vidět, grafickou interpretaci nemají všechny symboly. Symboly $[,], +$ a $-$ mají obvyklý význam, tj. zápis do zásobníku, načtení ze zásobníku, otočení v kladném směru a otočení v záporném směru.



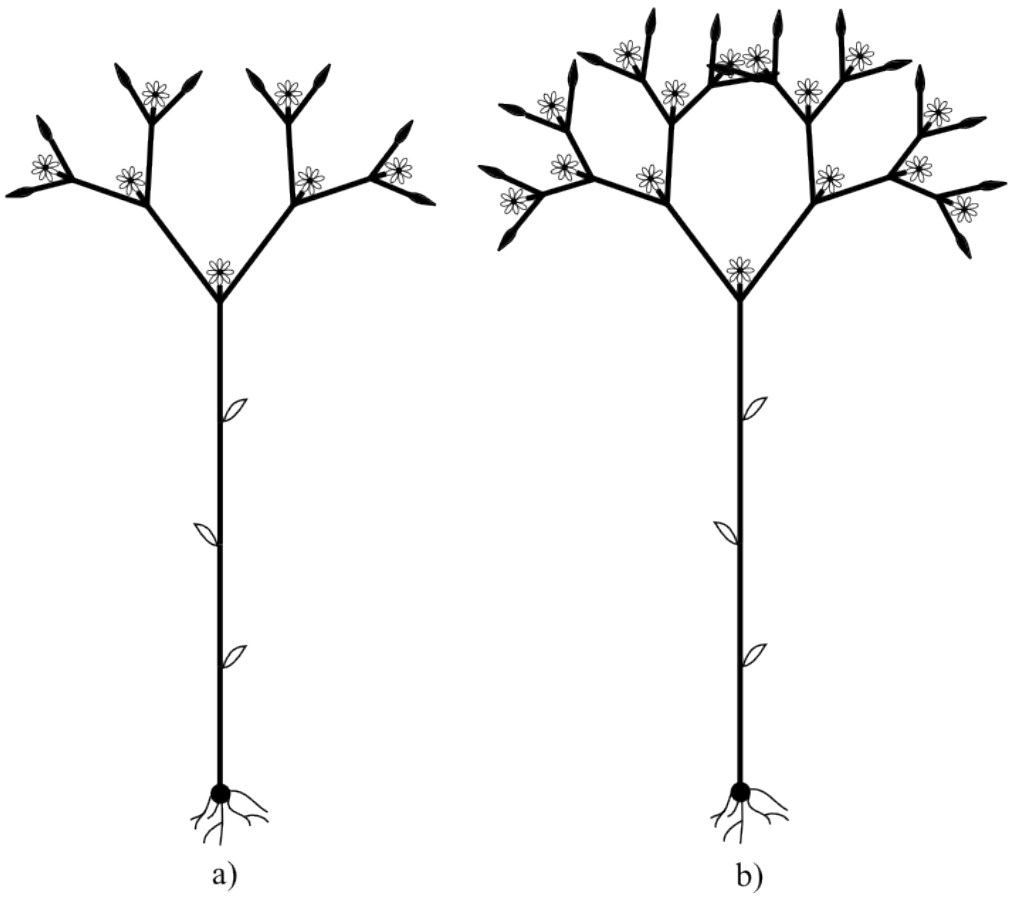
Obr. 1.4: Grafický význam symbolů

Množina parametrů X obsahuje dva parametry d a v. Parametr v slouží pro určení výšky, ale navyšuje se jen do doby, než dosáhne výšky 4. Pak již nezáleží na výšce. Pro jednoduchost výpočtů se tedy nepřepočítává. Parametry d slouží pro určení délky jedné části.



Obr. 1.5: Axiom a prvních šest iterací

Přepisovací pravidlo p se v L-systému aplikuje jen jednou, a to při přepsání axiomu s na posloupnost symbolů $Sf(1,1)$ - viz obr. 1.5 b) a zároveň určí vycházející hodnoty parametrů. Realizace přepisovacích pravidel q a r je závislá na levém kotextu. Přepisovací pravidlo q přepíše symbol $f(v,d)$ na posloupnost symbolů $F(v,d)[+L]f(v+1,d)$ pouze, pokud předním stojí posloupnost symbolů $-L]$. Podobně přepisovací pravidlo r přepíše symbol $f(v,d)$ na posloupnost symbolů $F(v,d)[-L]$ $f(v+1,d)$ pouze, pokud předním stojí posloupnost symbolů $+L]$. Obě přepisovací pravidla q a r se vykonají pouze, pokud hodnota parametru v je menší jak 4, aplikace obou pravidel je vidět na obr. 1.5 c) až e). Přepisovací pravidlo t přepíše symbol $f(v,d)$ na posloupnost symbolů $F(v,d_1)[-M][+M]K$ pouze, pokud hodnota parametru v přesahuje nebo se rovná 4, aplikování přepisovacího pravidla t je vidět na obr. 1.5 f). Přepisovací pravidlo u přepíše symbol M na posloupnost symbolů $F(v,\frac{2}{3}d_1)[-M][+M]K$ pouze, pokud předním stojí posloupnost symbolů $F(v,d_1)[\pm$.



Obr. 1.6: Sedmá a osmá iterace

Podmínka na levý kontext není uvedena kvůli podmínce na vypsání, ta by měla být splněna vždy, ale kvůli parametru d. Parametr d je závislý na parametru d symbolu F předcházejícím symbol M, který přepisujeme. Aplikace posledního přepisovacího pravidla u je vidět na obr. 1.5 g) a 1.6 a) i b).

Semínko rostliny je v příkladu 1.4 chápáno jako počáteční axiom. Růst stonku rostliny interpretují pravidla q a r a vykvetení rostliny interpretují pravidla t a u. Příklad 1.4 demonstruje, že realistický model nemusí být nutně složitý. Zapojení náhodného prvku do L-systémů nepřidá na složitosti, jak je poznat z příkladu 1.3. Ani aplikace L-systému do prostoru nepřidá na složitosti spíše na obsáhlosti.

Obr. 1.7: Ukázka L-systémem vytvořené rostliny – převzato z [4]

2 KONSTRUKCE MATEMATICKÉHO MODELU

L-systémy mají rozsáhlé využití v grafických aplikacích viz např. [2]. Ukázalo se, že pomocí L-systémů lze sestrojovat modely reálných rostlin - viz obr. 2.1, L-systémy mohou simulovat vliv gravitace a reakce na další vnější podněty. Takové modely však vyžadují rozsáhlý matematický aparát, který zde vzhledem k omezenému rozsahu práce nelze uvést. Omezíme se proto na modely využívající jen pojmy a vlastnosti popsané v předchozí kapitole.

Obr. 2.1: L-systém vygenerovaný pomocí programu, který byl součástí [1]

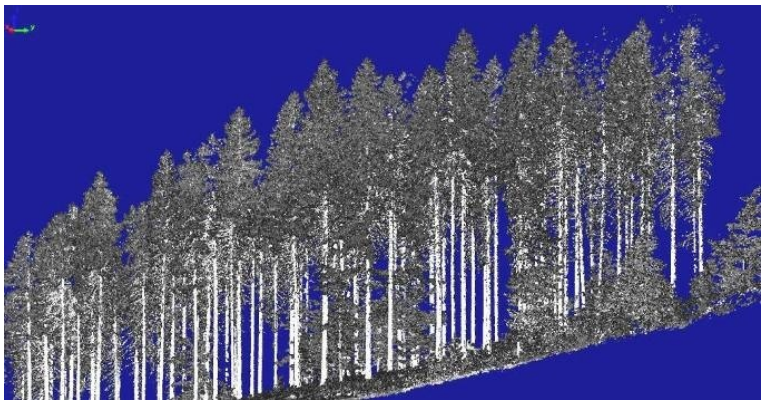
2.1 Popis problému

Stěžejní částí diplomové práce je návrh nedestruktivní metody umožňující popis struktury dospělého smrkového porostu. Strukturální parametry korunového zápoje budou sloužit pro parametrizaci virtuálního porostu v modelech radiativního transferu (model DART), tzn. modelech, které simulují interakce slunečního záření s libovolným vegetačním porostem. Využití modelů radiativního transferu slouží k vytvoření metodického postupu odhadu biochemických (např. obsah listového chlorofylu) a strukturálních parametrů (např. index listových ploch) z letových a družicových hyperspektrálních dat. Následující text v této kapitole je čerpán z [8] a z osobních

konzultací s pracovníky CVGZ a z jejich materiálů.

Pro laserové skenování byly vybrány dvě otevřené porostní stěny dospělého porostu smrku ztepilého (Picea abies (L.) Karst) v blízkosti experimentálního ekologického pracoviště Bílý Kříž provozovaného CVGZ (Moravsko-Slezské Beskydy, 18,54°E, 49,50°N, nadmořská výška 936 m).

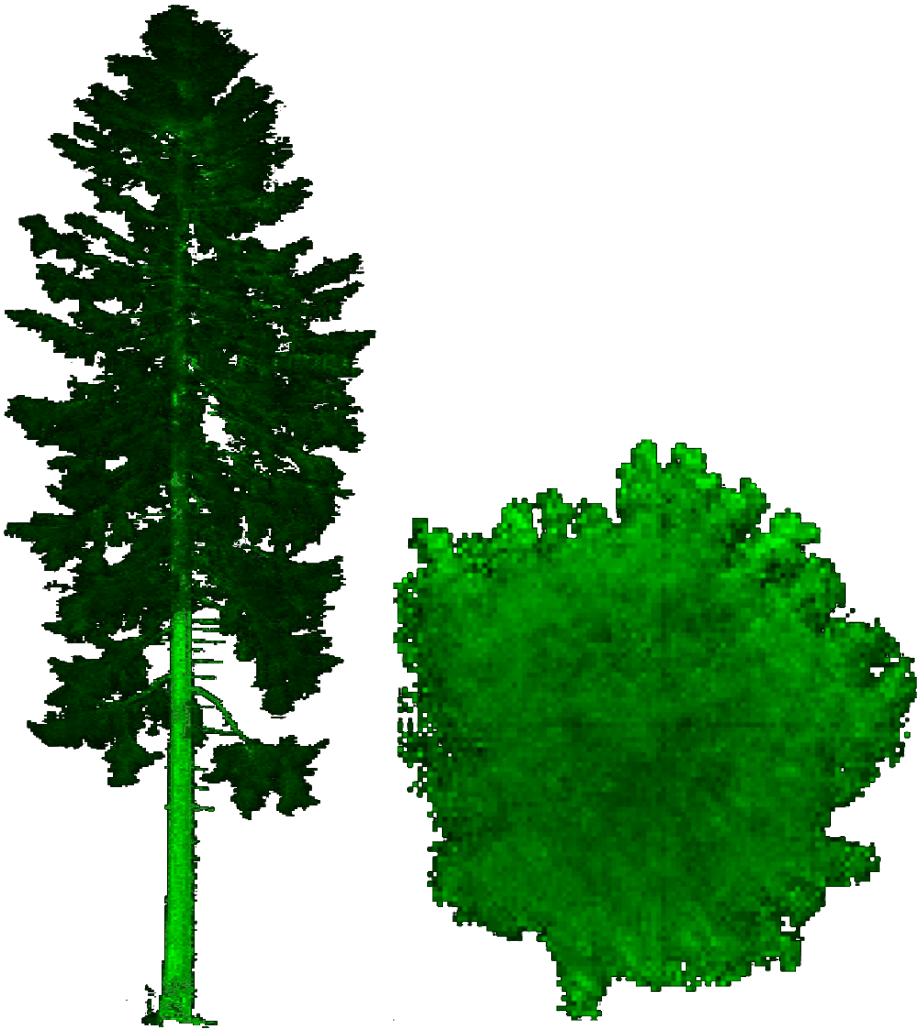
Laserové skenování na zvolených lokalitách bylo provedeno ve spolupráci s firmou GEOVAP s. r. o. přístrojem OPTECH Ilris-36D propojeným s digitálním fotoaparátem Canon EOS 35OD. Na obou lokalitách byl zaznamenán první odraz od porostní stěny. Data byla pořízena ze vzdálenosti cca 50 m a s rozestupem bodů 25 mm. Referenční body, které sloužily k transformaci do souřadnicového systému UTM 34N byly zaměřeny GPS systémem TOPCON GNSS. Ukázka laserových dat je na obr. 2.2



Obr. 2.2: Ukázka laserových dat

Manuálně bylo vybráno 27 individuálních stromů. Byla vylišena listová a dřevní biomasa na základě intenzity odraženého signálu ve vlnové délce 1500 nm. Pouze data odpovídající listové biomase byla dále statisticky zpracovány v prostředí ArcGIS. Statistické vyhodnocení rozmístěné biomasy v koruně stromu bylo použito pro výpočet parametrů, které charakterizují vertikální a horizontální rozmístění listového aparátu, jak je definováno v modelu DART. V programu PolyWorks IMView byly určeny základní parametry vybraných stromů (výška stromu, výška živé koruny, počet větví, zenitový úhel nasazení větve atd.). Data byla využita k určení půdorysu korun resp. k aproximaci půdorysu elipsou.

Analýza v ArcGIS sestávala jednak z určení řady statistických hodnot, jejichž výpočet byl modelován pomocí ModelBuilderu, a jednak v generování voxelového



Obr. 2.3: Voxelový model stromu

modelu stromu – viz obr. 2.3, tedy prostorového modelu rozložení listové hmoty uvnitř koruny (hrana voxelu byla stanovena na 0,1 m). Voxelový model byl dále doplněn o prostorový model sestávající z vrstev o výšce 1 m, které byly rozděleny na kruhové výseče se středovým úhlem 20° a dále na jednotlivá mezikruží s tloušťkou 0,2 m. Pro celý strom byl vypočten poloměr válce, jenž obsahuje 95 % všech bodů mračna, jakožto základní charakteristika průměru koruny. Pro voxelový model pak dalšími parametry jsou například:

- celkový počet neprázdných voxelů a průměrný počet bodů v těchto voxelech včetně směrodatné odchylky,
- průměrná a největší vzdálenost voxelů od středu stromu.

Pro výseče a vrstvy byly mimo jiné dále spočítány parametry:

- průměrný počet bodů v jednotlivých elementárních oblastech,
- největší vzdálenost pro každou výseč a každou vrstvu.

Díky takto naměřeným a zpracovaným datům je možno vytvořit virtuální model smrku pomocí L-systému. Data je nutné ještě předzpracovat, aby byla vhodná pro zavedení L-systému.

2.2 Poskytnutá data

Poskytnutá data se skládají z hodnot ručně naměřených na patnácti dospělých pokácených smrcích a ze zpracování laserových dat. Byly naměřeny tyto hodnoty: výška stromu, výška živé a mrtvé části koruny, počet větví, počet větví na patro, úhel větve, normované parametry $\alpha, \beta, \gamma, \kappa$, délka hlavní a vedlejší poloosy elipsy, úhel natočení, délka ročního přírůstku na větvi a úhel větvení. Význam naměřených hodnot popíšeme v následujícím textu.

Výška stromu, výška živé a mrtvé části koruny

Výška stromu byla měřena na nadzemní části stromu. Počínaje částí vycházející ze země až po špičku stromu. Živá část koruny je označení, jež se vztahuje na větve stromu vyrůstající z kmene a pokryté alespoň částečně zeleným jehličím. ***Výška živé části koruny*** tedy znamená vzdálenost mezi počátkem živé části koruny a špičkou stromu. Počátek živé části koruny se nachází na hranici živé a mrtvé části koruny. Mrtvá část koruny analogicky označuje větve vyrůstající z kmene a jež na sobě nemají již žádné zelené jehličí. ***Výška mrtvé části koruny*** je tedy vzdálenost od počátku růstu větví, od kořene stromu, až po hranici živé a mrtvé části koruny stromu. Rozdělení stromu na části – viz obr. 2.4.

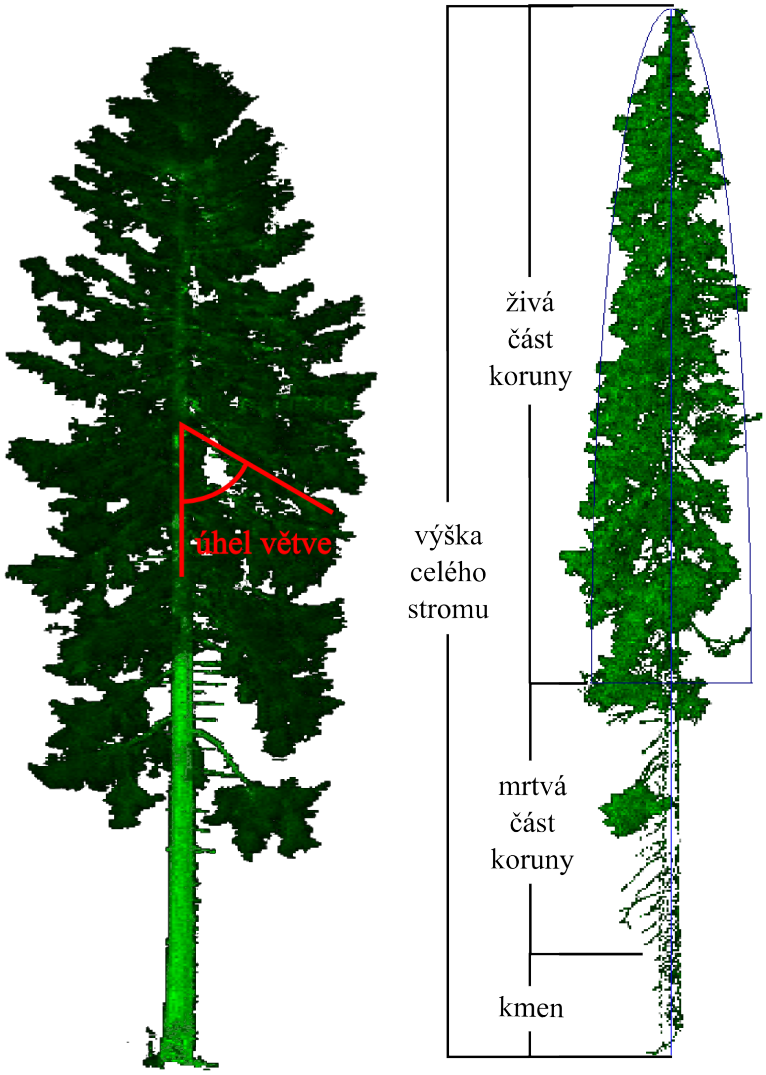
Počet větví

Do ***počtu větví*** se zahrnují jen větve vyrůstající z kmene stromu v živé části koruny. Měření proběhlo na dvou místech stromu na 2 m vysokém koláči. Počínaje v prvním případě špičkou stromu a v druhém případě hranicí živé a mrtvé části koruny. Výpočtem byl určen počet větví na celé živé části koruny.

Počet větví na patro

Větve na smrku jsou rozmístěny rovnoměrně po patrech. Každý rok vyroste nové patro a většinou je ***počet větví na jedno patro*** stejný. Liší se maximálně o jedna

od průměrného počtu větví na patro. Údaj o průměrném počtu větví byl naměřen na živých pokácených stromech.



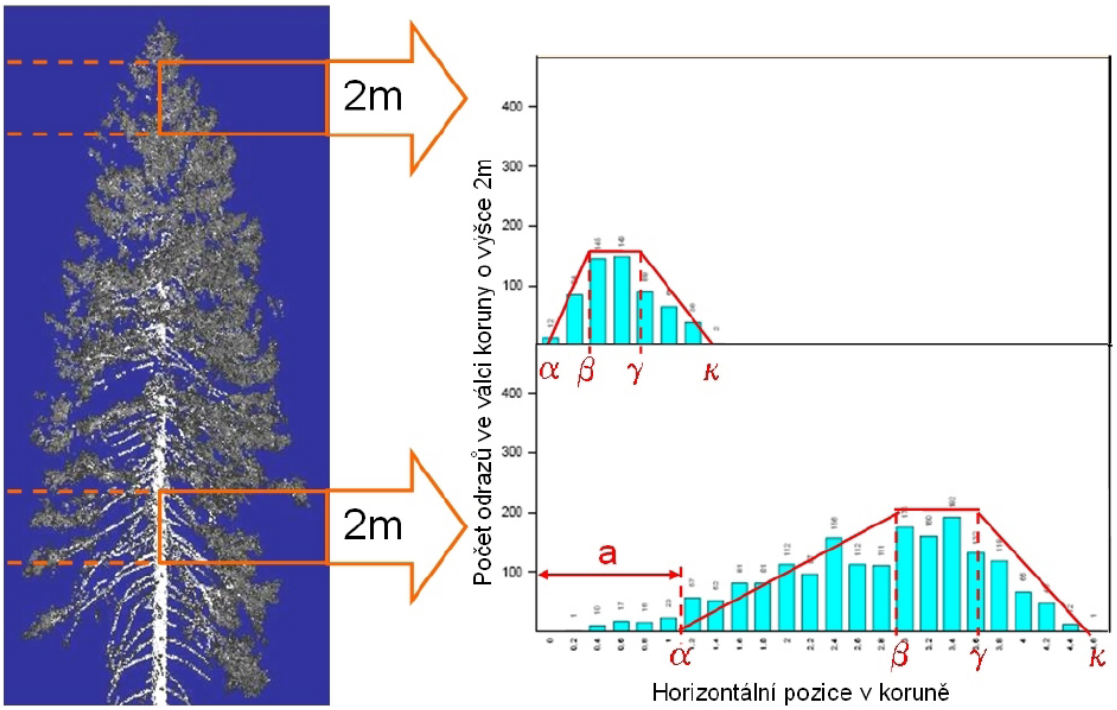
Obr. 2.4: Úhel větve a části stromu – použity voxelové modely z poskytnutých dat

Úhel větve

Úhel větve je úhel, který svírá část kmene zvlášť na živé a mrtvé části koruny. Bereme úhel, který svírá větev s částí kmene vedoucí ke kořeni stromu – viz obr. 2.4. Na živé části koruny se úhel měřil na třech místech: u špičky stromu, ve středu výšky živé části koruny a na rozmezí živé a mrtvé části koruny. U mrtvé části koruny se měřil úhel větví jen na jednom místě a aplikoval se na celou oblast mrtvé části koruny.

Normované parametry $\alpha, \beta, \gamma, \kappa$

Parametry $\alpha, \beta, \gamma, \kappa$ se týkají hustoty zeleně ve vzdálenosti od kmene stromu – viz obr. 2.5. Parametry byly měřeny na devíti rovnoměrně rozložených částech živé koruny každého stromu. Parametr α symbolizuje část větve, kde začíná růst zelené jehličí. β označuje část větve, kde začíná růst zeleného jehličí naplno. γ symbolizuje konec růstu zeleného jehličí naplno a κ je parametr, který znázorňuje konec růstu zeleného jehličí, což odpovídá konci větve. Platí, že $\alpha < \beta < \gamma < \kappa$. Poskytnutá data byla normovaná vůči výšce živé části koruny a maximální délce větve.



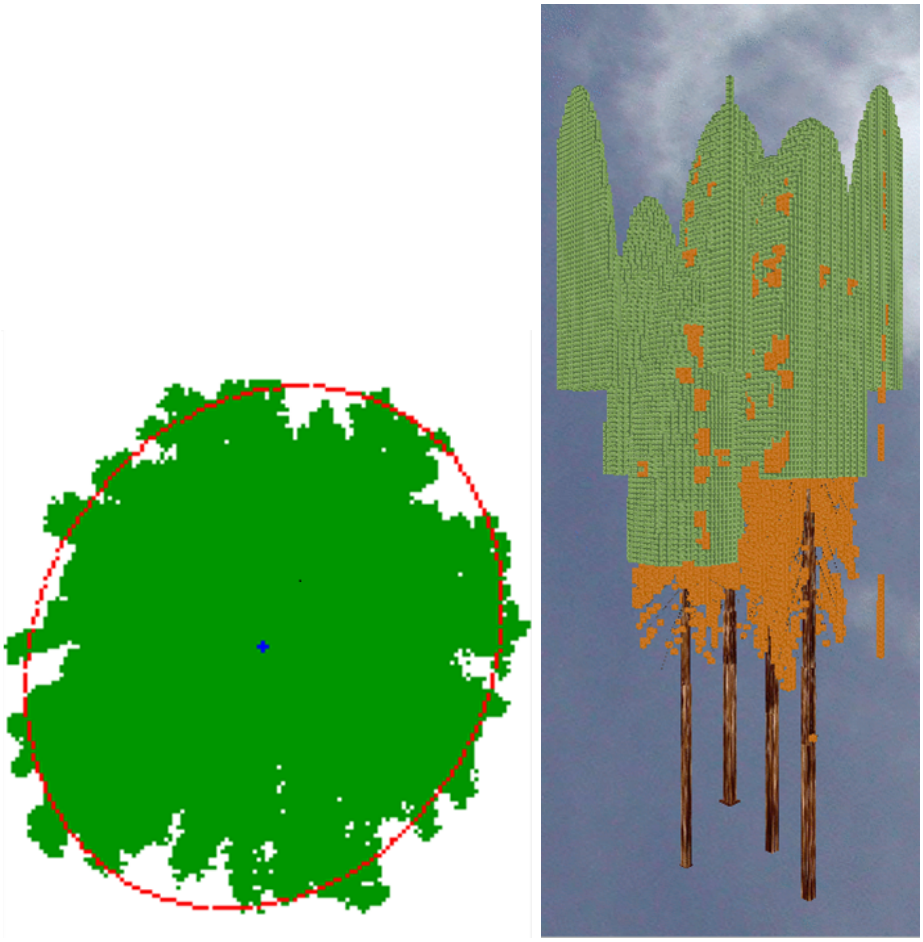
Obr. 2.5: Parametry $\alpha, \beta, \gamma, \kappa$ – obrázek byl poskytnut společně s daty na vytvoření modelu

Parametry elipsy a úhel natočení

Půdorys a nárys živé části koruny lze aproximovat dvěma elipsami – viz obr. 2.6. ***Půdorysná elipsa*** aproximuje rozhraní živé a mrtvé části koruny. ***Boční elipsa*** má jako ***hlavní poloosu*** výšku živé části koruny a ***vedlejší poloosu*** má umístěnou na rozhraní mrtvé a živé části koruny. Ve výsledku tedy stačí mít parametry půdorysné elipsy. Parametry boční elipsy se dají dopočítat z výšky a půdorysné elipsy. Půdorysná elipsa má navíc naměřený **úhel natočení** vůči globální soustavě souřadnic. Tyto parametry se týkají jen živé části koruny. Pro mrtvou část koruny se

hranice, kam dosahují mrtvé větve, určí válcem, který má za podstavu půdorysnou elipsu a za výšku výšku mrtvé části koruny.

Pozn.: *Přestože jsme uvedli jen jednu boční elipsu, je jich nekonečně mnoho. Boční elipsy mají všechny stejnou hlavní poloosu, ale vedlejší poloosa se mění. Určuje ji vzdálenost mezi středem půdorysné elipsy a libovolným bodem na půdorysné elipse. Boční elipsu myslíme vždy tu, v níž leží větev, se kterou pracujeme nebo aktuální boční pohled.*

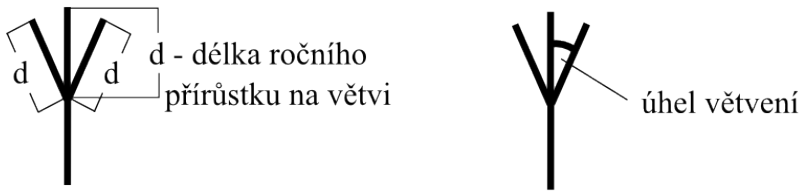


Obr. 2.6: Půdorysná a boční elipsa – obrázky byly poskytnuty společně s daty na vytvoření modelu

Délka ročního přírůstku na větvi a úhel větvení

Ročně se na konci každé větve vytvoří nové přírůstky větve, není pravidlem, že se na každém konci větve musí vytvořit nový přírůstek vždy. Může se stát, že na konci větve nevyroste jeden rok nic. Přírůstky mohou být maximálně tři, každý do jiného

směru. ***Délka ročního přírůstku na větvi*** je délka jednoho takového dílku – viz obr. 2.7. Délka ročního přírůstku na větvi by měla být konstantní pro jednotlivé stromy.



Obr. 2.7: Délka ročního přírůstku a úhel větvení

Při ročním přírůstku mohou větve růst z jednoho konce pouze do tří směrů, které leží v jedné rovině. Přírůstek může pokračovat v prodloužení větve a nebo se rozvětvit do dvou směrů, kladného a záporného. ***Úhel větvení*** určuje, o kolik se odchýlí nový přírůstek do kladného nebo záporného směru od původního směru růstu – viz obr. 2.7.

2.3 Zpracování dat

Data byla naměřena na patnácti vzorcích, ale model slouží k vygenerování jednoho smrku. Data tudíž musíme zpracovat před použitím na tvorbu modelu. Většinu parametrů vztáhneme na výšku celého stromu, takže parametry jsou ve tvaru výška celého stromu násobena poměrnou částí parametru k výšce celého stromu. Podle naměřených dat vypočítáme rozmezí pro náhodnou volbu daného parametru. Náhodnost parametrů zaručuje přirozenější vzhled modelu.

Pro model smrku dopočítáme funkci ϑ charakterizující závislost úhlu živé větve na výšce živé části koruny. Hodnoty byly měřeny na třech místech na živé části koruny a byly aproximovány lineární funkcí. Pro parametry p, q tedy platí

$$\vartheta = pv + q, \tag{2.1}$$

kde v je výška, ve které větev vyrůstá z kmene.

Dalším parametrem je maximální délka větve. Díky elipsám známe koncové body větví, ale jelikož nevyrůstají kolmo ke kmeni stromu, není maximální délka větve totéž jako hranice stromu vymezená elipsami. Nejprve získáme vztah pro výpočet

vedlejší poloosy boční elipsy. Vycházíme z parametrického předpisu pro elipsu:

$$x = a_p \cdot cos\psi, \tag{2.2}$$

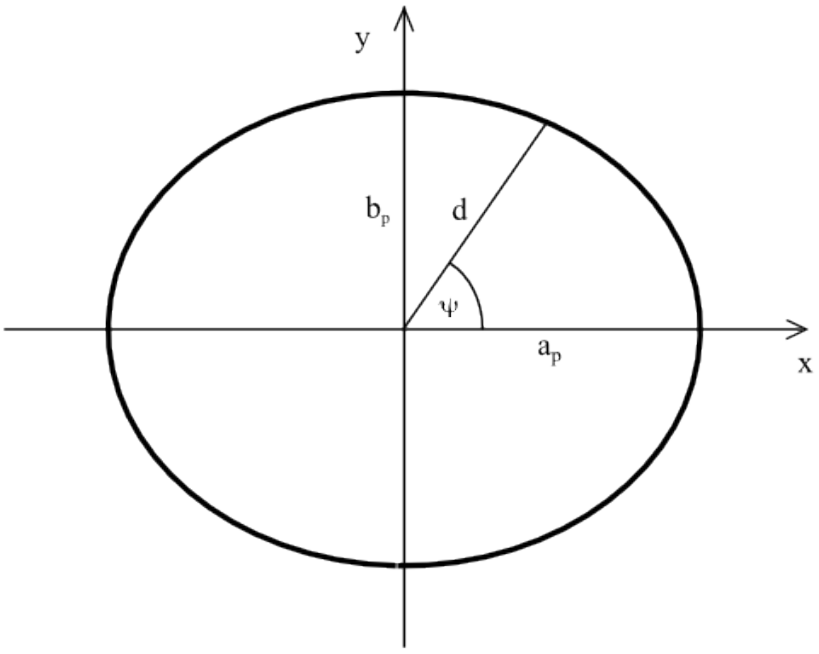
$$y = b_p \cdot sin\psi, \tag{2.3}$$

kde a_p je délka hlavní poloosy půdorysné elipsy, b_p je délka vedlejší poloosy půdorysné elipsy a ψ je natočení od osy x. Soustavu souřadnic umístíme do roviny půdorysné elipsy tak, že hlavní poloosa leží na ose x a vedlejší poloosa leží na ose y. Umístění elipsy a parametry související s výpočty jsou na obr. 2.8. Pro vzdálenost d od počátku, kde máme umístěn střed kmene, do libovolného bodu elipsy použijeme vztah:

$$d = \sqrt{x^2 + y^2} \tag{2.4}$$

Po dosazení:

$$d = \sqrt{a_p^2 cos^2\psi + b_p^2 sin^2\psi}. \tag{2.5}$$



Obr. 2.8: Znázornění parametrů na půdorysné elipse

Z rovnice 2.5 dostaneme vedlejší poloosu boční elipsy b_b, takže $b_b = d$. Výška živé části koruny je hlavní poloosou boční elipsy a_b. Výšku v, ve které vyrůstá větev z kmene, zjistíme ze známého počtu živých větví a z průměrného počtu větví na jedno patro. Z poskytnutých dat tedy dopočítáme výšku, ve které vyrůstá větev

z kmene, a úhel větve ϑ již z poskytnutých dat také známe. Soustavu souřadnic umístíme do roviny boční elipsy, ve které leží větev. Její hlavní poloosa leží na ose x a vedlejší poloosa leží na ose y. Umístění elipsy a parametry související s výpočty jsou na obr. 2.9. Maximální délku větve m zjistíme z průniku přímky obsahující větev a boční elipsy. Přímka je určena bodem $[v; 0]$, ve kterém vyrůstá větev z kmene stromu a směrnicí, která je dána úhlem, pod kterým větev vyrůstá z kmene stromu. Rovnice přímky je:
Pro $\vartheta > 90°$:

$$y = (x - v)tan(180° - \vartheta), \tag{2.6}$$

pro $\vartheta < 90°$:

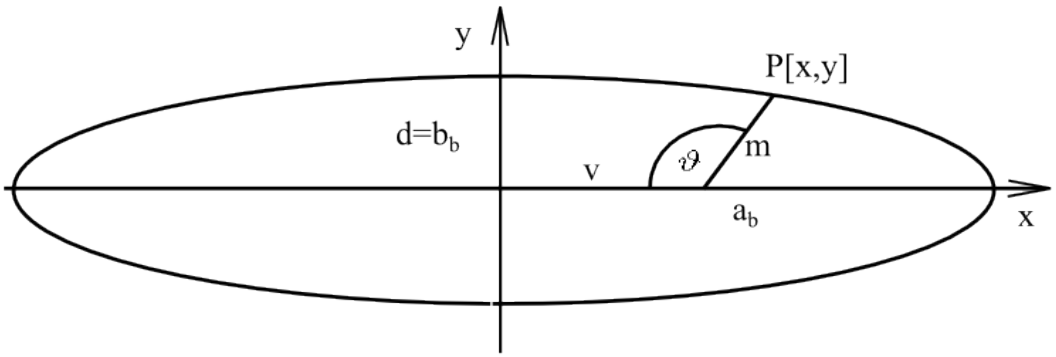
$$y = (v - x)tan(\vartheta). \tag{2.7}$$

Rovnici elipsy použijeme ve tvaru:

$$\frac{x^2}{a_b^2} + \frac{y^2}{b_b^2} = 1. \tag{2.8}$$

Rovnice 2.6, 2.7 a 2.8 určují průsečík $P[x; y]$ přímky s elipsou. Maximální délka větve m je pak dána vztahem:

$$m = \sqrt{(v - x)^2 + y^2}. \tag{2.9}$$



Obr. 2.9: Znázornění parametrů na boční elipse

Pro parametry α, β, γ je potřeba určit rovnici závislosti parametrů na výšce v a maximální délce větve m. Experimentálně bylo zjištěno, že tuto závislost nejlépe

vystihují následující lineární funkce:

$$\alpha = (-0.1495\frac{v}{m} + 0.1925)m \tag{2.10}$$

$$\beta = (-0.3387\frac{v}{m} + 0.4345)m \tag{2.11}$$

$$\gamma = (-0.3379\frac{v}{m} + 0.7546)m \tag{2.12}$$

Nyní máme k dispozici všechny potřebné parametry stromu, abychom byli schopni vytvořit algoritmus pro vytvoření virtuálního modelu a vytvoření L-systému.

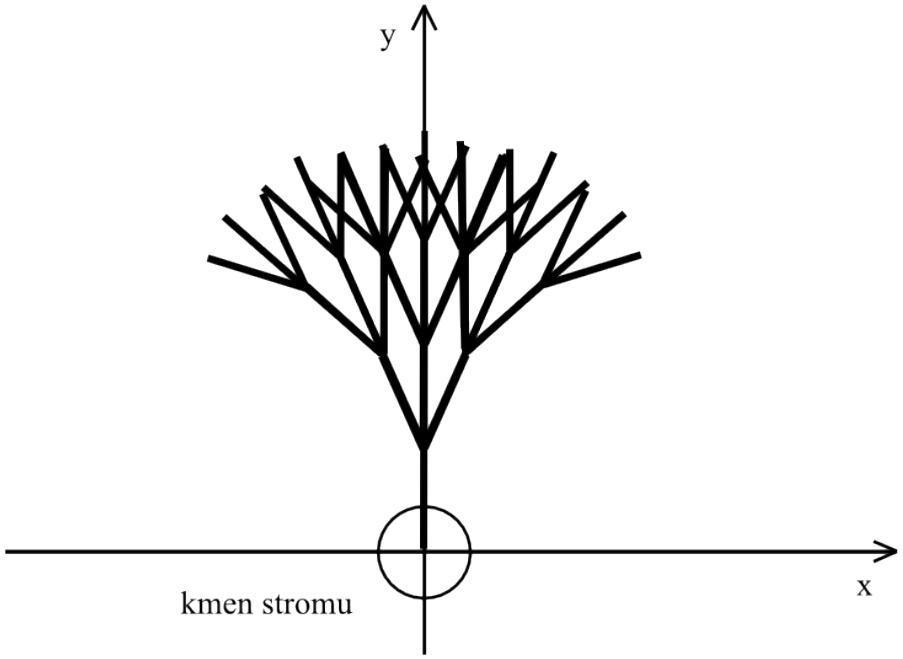
2.4 Tvorba L-systému

L-systémy slouží pouze k vygenerování jednotlivých větví. L-systém využívá tyto parametry: délka ročního přírůstku na větvi, maximální délka větve a úhel větvení φ.

Jedna iterace v L-systému odpovídá růstu během jednoho roku. Během tvorby L-systému chápeme větev jako úsečku, tj. předpokládáme její nulový průměr. Větev se může větvit maximálně do tří směrů: v prodloužení, do kladného a do záporného směru. Větev se větví pouze do roviny - nikdy nejsou přírůstky pod nebo nad větev. Rovina větve prochází bodem, ze kterého vyrůstá větev z kmene, a je kolmá na rovinu úhlu ϑ, který svírá větev a kmen – viz obr. 2.4. Větev se nejdříve vytvoří v rovině a pak až se umístí do prostoru. Hlavní větev, ta jež vyrůstá přímo z kmene, je vždy umístěna na ose y. Umístění do souřadné soustavy je vidět na obr. 2.10 Větev před umístěním do souřadné soustavy stromu projde dvěma transformacemi. Nejprve otočíme větev kolem osy kmene stromu o úhel ψ a podruhé natočíme větev vůči hlavní ose kmene stromu o úhel větve ϑ. Transformace a reálná podoba větví se nemodelují v L-systému samotném. Realizují se až vizualizací, která bude popsána v následující podkapitole. Transformace jsou vidět na obrázku 3.6

L-systém pro větev smrku zavedeme následovně.

- L-systém $\mathcal{L}$ je uspořádaná čtveřice $\mathcal{L} = (\Sigma, P, \sigma, X)$,
- množina terminálů $T = \{V, +, -, [,]\}$ a množina neterminálů $N = \{v\}$, prázdný symbol není v L-systému využit,
- axiom $\sigma = v$,
- množina parametrů $X = \{A, B, z, i\}$, kde $A = [a_x, a_y]$, $B = [b_x, b_y]$, $z, i \in \mathbb{N}$,
- množina přepisovacích pravidel $P = \{p_1, p_2, p_3, p_4\}$.



Obr. 2.10: Umístění větve do souřadné soustavy

Význam znaků abecedy Σ – viz obr. 2.11:

- Znaky V a v reprezentují posun vpřed a konstrukci části větve o délce ročního přírůstku na větvi, v prvním případě se znak už dále nepřepisuje a ve druhém se znak v další iteraci přepíše podle přepisovacích pravidel.
- Znaky [a] reprezentují zápis současných souřadnic do paměti zásobníku a načtení odpovídající pozice a natočení ze zásobníku.
- Znaky + a − reprezentují otočení o daný úhel do kladného či záporného směru.

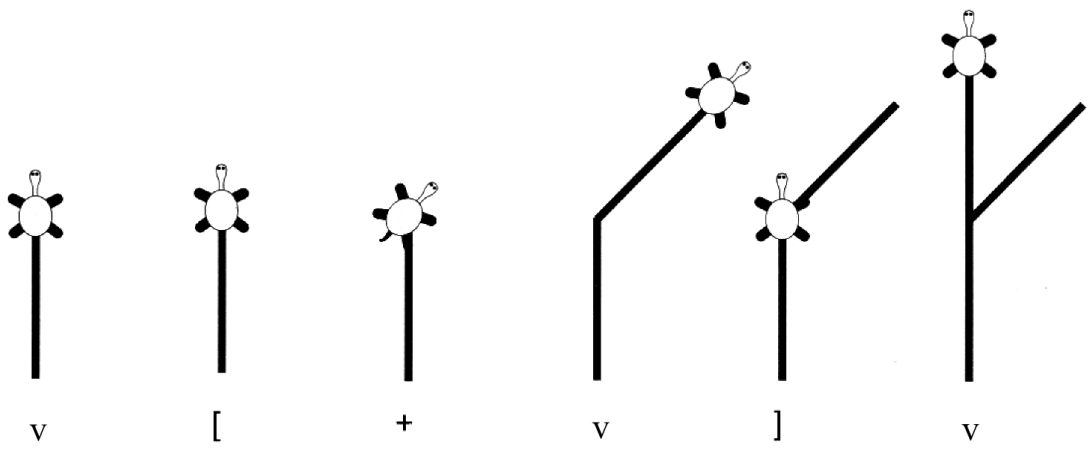
Význam parametrů množiny X:

- Parametr A reprezentuje počáteční bod části větve ve tvaru $A = [a_x, a_y]$, kde a_x je x-ová a a_y je y-ová souřadnice bodu A.
- Parametr B reprezentuje koncový bod části větve neboli vrchol aktuálního kužele větve ve tvaru $B = [b_x, b_y]$, kde b_x je x-ová a b_y je y-ová souřadnice bodu B.
- Parametr z reprezentuje rok, ve kterém větev vyrostla.
- Parametr i reprezentuje index kužele, kterému tato větev náleží.

Přepisovací pravidla množiny P:

- Přepisovací pravidlo p_1 se týká přírůstku na intervalu $\langle 0, \alpha \rangle$.

- Přepisovací pravidlo p_2 se týká přírůstku na intervalu $\langle \alpha, \beta \rangle$.
- Přepisovací pravidlo p_3 se týká přírůstku na intervalu $\langle \beta, \gamma \rangle$.
- Přepisovací pravidlo p_4 se týká přírůstku na intervalu $\langle \gamma, d_{max} \rangle$.



Obr. 2.11: Význam jednotlivých znaků – převzato z [1]

Nyní musíme zavést proměnnou i_m tak, aby odpovídala nejvýše dosaženému indexu i pro L-systém větve. Proměnná i_m se bude měnit vždy, když se nějaká větev rozroste kladného či záporného směru. Přepisovací pravidla mají tvar:

$$p_1 : \mathbf{v}(A, B, z, i) : b_y \in \langle 0, \alpha \rangle \rightarrow$$

$$\mathbf{V}(A, B, z, i)\ \mathbf{v}(B, B + d, z + 1, i) : \mu_{p_1},$$

$$p_2 : \mathbf{v}(A, B, z, i) : b_y \in \langle \alpha, \beta \rangle \rightarrow$$

$$\mathbf{V}(A, B, z, i)[+\mathbf{v}(B, B + d_\varphi, z + 1, i_m + 1)][-\mathbf{v}(B, B + d_\varphi, z + 1, i_m + 2)]$$
$$\mathbf{v}(B, B + d, z + 1, i) :\ \mu_{p_{21}},$$

$$\mathbf{V}(A, B, z, i)[+\mathbf{v}(B, B+d_\varphi, z+1, i_m+1)][-\mathbf{v}(B, B+d_\varphi, z+1, i_m+2)] : \mu_{p_{22}},$$

$$\mathbf{V}(A, B, z, i)\ [+\mathbf{v}(B, B + d_\varphi, z + 1, i_m + 1)\ ]\ \mathbf{v}(B, B + d, z + 1, i) : \mu_{p_{23}},$$

$$\mathbf{V}(A, B, z, i)\ [-\mathbf{v}(B, B + d_\varphi, z + 1, i_m + 1)\ ]\ \mathbf{v}(B, B + d, z + 1, i) : \mu_{p_{24}},$$

$$\mathbf{V}(A,B,z,i)\ \mathbf{v}(B,B+d,z+1,i) : \mu_{p_{25}},$$

$$\mathbf{V}(A,B,z,i)\ [-\mathbf{v}(B,B+d_\varphi,z+1,i_m+1)\ ] : \mu_{p_{26}},$$

$$\mathbf{V}(A,B,z,i)\ [+\mathbf{v}(B,B+d_\varphi,z+1,i_m+1)\ ] : \mu_{p_{27}},$$

$$\mathbf{v}(A,B,z,i) : \mu_{p_{28}},$$

$p_3 : \mathbf{v}(A,B,z,i) : b_y \in \langle \beta,\gamma \rangle \rightarrow$

$$\mathbf{V}(A,B,z,i)[+\mathbf{v}(B,B+d_\varphi,z+1,i_m+1)][-\mathbf{v}(B,B+d_\varphi,z+1,i_m+2)]$$
$$\mathbf{v}(B,B+d,z+1,i) : \ \mu_{p_3},$$

$p_4 : \mathbf{v}(A,B,z,i) : b_y \in \langle \gamma,d_{max} \rangle \rightarrow$

$$\mathbf{V}(A,B,z,i)[+\mathbf{v}(B,B+d_\varphi,z+1,i_m+1)][-\mathbf{v}(B,B+d_\varphi,z+1,i_m+2)]$$
$$\mathbf{v}(B,B+d,z+1,i) : \ \mu_{p_{41}},$$

$$\mathbf{V}(A,B,z,i)[+\mathbf{v}(B,B+d_\varphi,z+1,i_m+1)][-\mathbf{v}(B,B+d_\varphi,z+1,i_m+2)] : \mu_{p_{42}},$$

$$\mathbf{V}(A,B,z,i)\ [+\mathbf{v}(B,B+d_\varphi,z+1,i_m+1)\ ]\ \mathbf{v}(B,B+d,z+1,i) : \mu_{p_{43}},$$

$$\mathbf{V}(A,B,z,i)\ [-\mathbf{v}(B,B+d_\varphi,z+1,i_m+2)\ ]\ \mathbf{v}(B,B+d,z+1,i) : \mu_{p_{44}},$$

$$\mathbf{V}(A,B,z,i)\ \mathbf{v}(B,B+d,z+1,i) : \mu_{p_{45}},$$

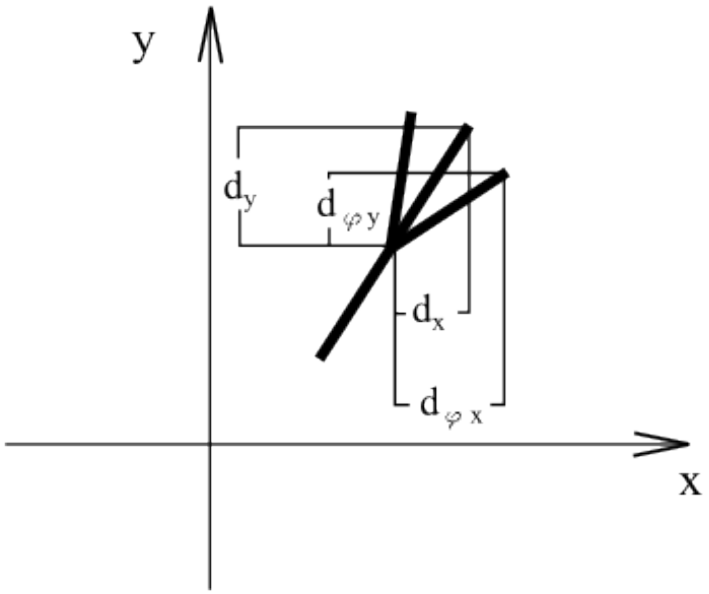
$$\mathbf{V}(A,B,z,i)\ [-\mathbf{v}(B,B+d_\varphi,z+1,i_m+1)\ ] : \mu_{p_{46}},$$

$$\mathbf{V}(A,B,z,i)\ [+\mathbf{v}(B,B+d_\varphi,z+1,i_m+1)\ ] : \mu_{p_{47}},$$

$$\mathbf{v}(A,B,z,i) : \mu_{p_{48}},$$

Parametry $d_\varphi = [d_{\varphi x}, d_{\varphi y}]$ a $d = [d_x, d_y]$ jsou při každém použití jiné, proto zde musíme uvést, jak se k těmto hodnotám přistupuje. Hodnotu d chápeme jako přírůstek na délce v prodloužení větve. Tudíž d chápeme jako $d = \sqrt{d_x^2 + d_y^2}$, kde d_x je délkový přírůstek na ose x a d_y na ose y a d odpovídá délce jedné části větve – viz obr. 2.12. Jelikož při každém větvení se mění směr prodloužení větve, modifikuje se vždy přírůstek na ose x i y. Kdybychom do přepisovacích pravidel zavedli i tyto

transformace, byly by značně rozsáhlé a nepřehledné, a proto je tento přírůstek napsaný jen symbolicky. Hodnota d_φ má obdobný význam, ale ne ve směru prodloužení, ale ve směru větvení do kladného či záporného směru o úhel φ – viz obr. 2.12. Vždy před použitím parametru d_φ se vygeneruje náhodný úhel φ aplikovaný ve větvení. Pak se podle znaku + nebo − rozhodne o směru růstu větve.



Obr. 2.12: Znázornění parametrů d a d_φ

Přepisovací pravidla p_1 až p_4 jsou aplikována pouze na konkrétní část větve vymezené parametry α, β a γ. Jelikož pomocí pravidel musíme vyjádřit hustotu té části větve resp. hustotu zeleně na větvi, musí se na intervalech $\langle \alpha, \beta \rangle$ a $\langle \gamma, d_{max} \rangle$ mocnost pravidel měnit v závislosti na délce. Pro intervaly $\langle 0, \alpha \rangle$ a $\langle \beta, \gamma \rangle$ je pravidlo jen jedno a jejich mocnost je rovna 1.

Na intervalu $\langle 0, \alpha \rangle$ roste větev pouze dopředu, protože na tomto intervalu žádná zeleň neroste. Oproti tomu na intervalu $\langle \beta, \gamma \rangle$ roste větev naplno do všech tří směrů. Na zbývajících intervalech je postupný nárůst nebo pokles hustoty větvení a má lineární průběh. Jelikož můžeme rozdělit růst větví na čtyři skupiny podle počtu nových větví v jednom větvení, mocnosti pravidel rozdělíme na čtyři skupiny. Mocnosti se nebudou navzájem vylučovat, ale jedna bude mít vždy větší mocnost než zbylé tři. Dosáhneme toho pomocí zavedení funkcí příslušné mocnosti každé skupině

závislé na vzdálenosti od kmene. Funkce zavedeme následovně:

pro $\mu_{p_{28}}$:

$$f_1 = \begin{cases} 1 - \frac{(v_k - \alpha)}{0{,}25(\beta - \alpha)} & \text{pro } v_k \in \langle \alpha, \alpha + 0.25(\beta - \alpha) \rangle, \\ 0 & \text{jinak,} \end{cases}$$

pro $\mu_{p_{25}}, \mu_{p_{26}}$ a $\mu_{p_{27}}$:

$$f_2 = \begin{cases} \frac{v_k - \alpha}{0{,}25(\beta - \alpha)} & \text{pro } v_k \in \langle \alpha, \alpha + 0.25(\beta - \alpha) \rangle, \\ \frac{-v_k + \alpha + 0{,}75(\beta - \alpha)}{0{,}5(\beta - \alpha)} & \text{pro } v_k \in \langle \alpha + 0.25(\beta - \alpha), \alpha + 0.75(\beta - \alpha) \rangle, \\ 0 & \text{jinak,} \end{cases}$$

pro $\mu_{p_{22}}, \mu_{p_{23}}$ a $\mu_{p_{24}}$:

$$f_3 = \begin{cases} \frac{v_k - \alpha - 0{,}25(\beta - \alpha)}{0{,}5(\beta - \alpha)} & \text{pro } v_k \in \langle \alpha + 0.25(\beta - \alpha), \alpha + 0.75(\beta - \alpha) \rangle, \\ \frac{-v_k + \beta}{0{,}75(\beta - \alpha)} & \text{pro } v_k \in \langle \alpha + 0.25(\beta - \alpha), \alpha + 0.75(\beta - \alpha) \rangle, \\ 0 & \text{jinak,} \end{cases}$$

pro $\mu_{p_{21}}$:

$$f_4 = \begin{cases} 1 + \frac{(v_k - \beta)}{0{,}25(\beta - \alpha)} & \text{pro } v_k \in \langle \alpha, \alpha + 0.25(\beta - \alpha) \rangle, \\ 0 & \text{jinak,} \end{cases}$$

pro $\mu_{p_{41}}$:

$$f_5 = \begin{cases} 1 - \frac{(v_k - \gamma)}{0{,}25(d_{max} - \gamma)} & \text{pro } v_k \in \langle \gamma, \gamma + 0.25(d_{max} - \gamma) \rangle, \\ 0 & \text{jinak,} \end{cases}$$

pro $\mu_{p_{42}}, \mu_{p_{43}}$ a $\mu_{p_{44}}$:

$$f_6 = \begin{cases} \frac{v_k - \gamma}{0{,}25(d_{max} - \gamma)} & \text{pro } v_k \in \langle \gamma, \gamma + 0.25(d_{max} - \gamma) \rangle, \\ \frac{-v_k + \gamma + 0{,}75(d_{max} - \gamma)}{0{,}5(d_{max} - \gamma)} & \text{pro } v_k \in \langle \gamma + 0.25(d_{max} - \gamma), \gamma + 0.75(d_{max} - \gamma) \rangle, \\ 0 & \text{jinak,} \end{cases}$$

pro $\mu_{p_{45}}, \mu_{p_{46}}$ a $\mu_{p_{47}}$:

$$f_7 = \begin{cases} \frac{v_k - \gamma - 0{,}25(d_{max} - \gamma)}{0{,}5(d_{max} - \gamma)} & \text{pro } v_k \in \langle \gamma + 0.25(d_{max} - \gamma), \gamma + 0.75(d_{max} - \gamma) \rangle, \\ \frac{-v_k + d_{max}}{0{,}75(d_{max} - \gamma)} & \text{pro } v_k \in \langle \gamma + 0.25(d_{max} - \gamma), \gamma + 0.75(d_{max} - \gamma) \rangle, \\ 0 & \text{jinak,} \end{cases}$$

pro $\mu_{p_{48}}$:

$$f_8 = \begin{cases} 1 + \frac{(v_k - d_{max})}{0{,}25(d_{max} - \gamma)} & \text{pro } v_k \in \langle \gamma, \gamma + 0.25(d_{max} - \gamma) \rangle, \\ 0 & \text{jinak,} \end{cases}$$

kde v_k je vzdálenost od kmene stromu. Funkcemi f jsou vyjádřeny mocnosti pouze pro pravidla tvořící určitý počet větví. Např. mocnosti $\mu_{p_{45}}, \mu_{p_{46}}$ a $\mu_{p_{47}}$ jsou chápány jako jedna mocnost. Pokud tedy dojde na jeden z těchto případů, musí se vybírat ještě z dalších tří pravidel pro daný počet větví. Pravděpodobnost výběru jednoho z těchto tří pravidel je rovna $\frac{1}{3}$.

3 REALIZACE

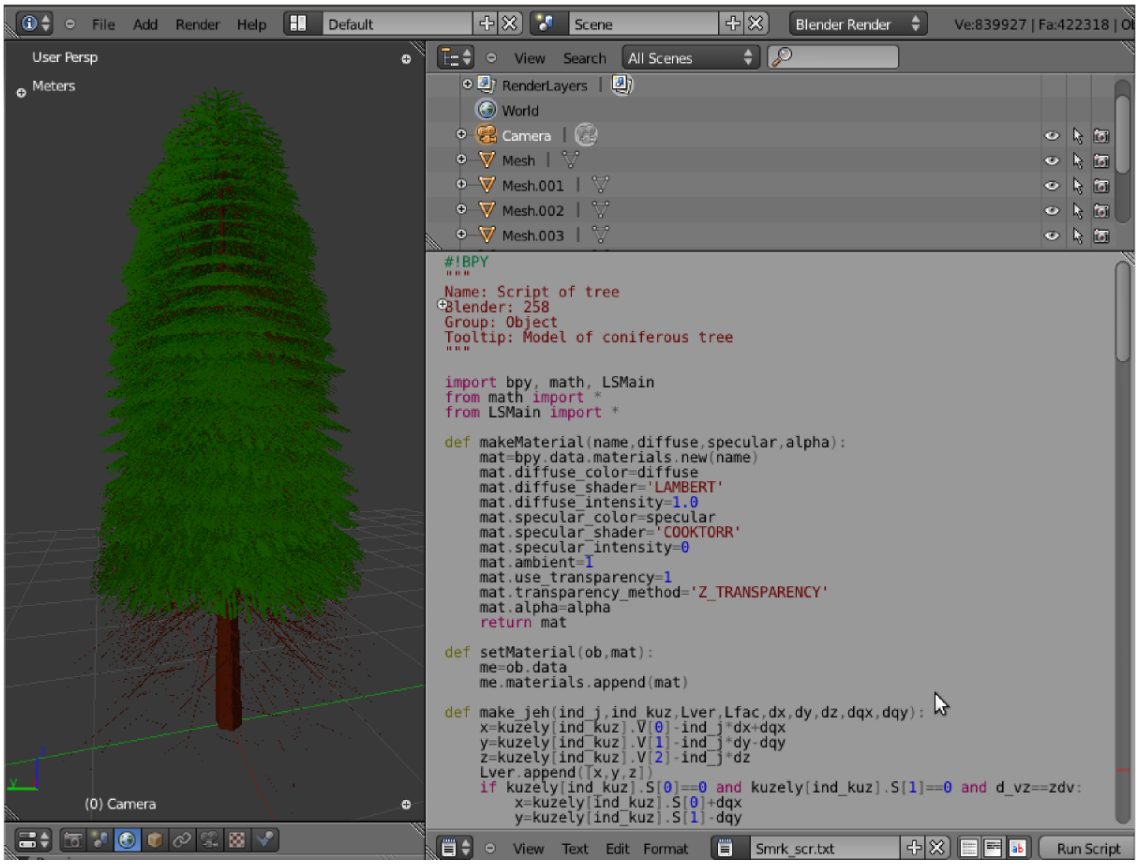
3.1 Volba programovacího prostředí

Zadání problému nepožadovalo konkrétní programátorské prostředí, určovalo jen formát výstupu. Volba prostředí však byla pro řešení zcela zásadní. Naskytovalo se mnoho grafických softwarů s možným exportováním modelu ve formátu, který byl požadován. Posuzované vlastnosti grafického softwaru byly dostupnost, rozšířenost, možnost implementace vlastních skriptů a programovací jazyk pro tvorbu těchto skriptů.

Dostupnost je důležitá z hlediska placení licencí a následného distribuování. Rozšířenost vyplývá z různých vlastností softwaru. Z hlediska dostupnosti je důležitá přítomnost různých návodů, příkladů a faktorů, které mají vliv na zvládnutí softwaru. Možnost implementace vlastních skriptů je hlavní z hlediska tvorby modelu. Jelikož chceme generovat L-systém a potažmo celý strom pomocí algoritmu, je vhodné mít ho jak implementovat přímo do grafického softwaru bez nutnosti využití dalšího prostředku na převod kódu. Programovací jazyk je vhodné volit podle toho, jaké jsou nynějších schopnosti programátora, případně podle náročnosti naučení programování v tomto jazyce.

Jeden z návrhů od CVGZ byl grafický software Blender, jehož výstup je CVGZem ověřen a vyhovuje jeho požadavkům. Po prozkoumání charakteristik Blenderu se zjistilo, že vyhovuje všem požadovaným vlastnostem. Blender je velice rozšířený grafický software pro tvorbu 3D modelů, animovaných videí nebo počítačových her. Existuje pro něj již mnoho předpřipravených skriptů pro implementaci běžných prvků přírody, lidských prvků a mnohé další. Mezi nimi byly i skripty na tvorbu stromů. Objevil se návrh na modifikaci již hotového skriptu tak, aby tvořil model smrku podle zadaných parametrů. Tento návrh byl zamítnut, protože úprava kódu by byla ve výsledku stejně ne-li více náročná něž tvorba kódu od začátku podle potřeb našeho modelu. Více o Blenderu lze nalézt v [5]. Pro vytvoření modelu byla použita verze 2.58.0. Ukázka prostředí Blenderu je vidět na obr. 3.1.

Skripty do Blenderu se implementují velice jednoduše a jsou psány v programovacím jazyce Python. Tento jazyk bohužel nebyl předmětem výuky, jeho zvládnutí však nebylo náročné. Python je vysoce výkonný programovací jazyk používající efektivní vysokoúrovňové datové typy, přičemž jednoduše a elegantně řeší otázku objektově orientovaného programování. Jeho syntaxe a dynamické typy spolu s interpretováním kódu dotváří pověst ideálního nástroje pro psaní skriptů a rychlý



Obr. 3.1: Ukázka programovacího prostředí Blender

vývoj aplikací (Rapid Application Development, RAD). Samotný interpretr jazyka je spustitelný na velkém množství platforem. Více informací lze nalézt v [6] nebo [7]. Blender má implementovaný Python verze 3.2, proto jsme i skripty tvořily v této verzi Pythonu. Dobré je také vědět, že knihovna pro matematické funkce počítá s radiány stejně tak i grafické knihovny Blenderu. Ukázka programovacího prostředí Python (v editoru Pype) je vidět na obr. 3.2.

Instalace obou programů není složitá, ale je třeba dbát na to, aby byla instalována verze Pythonu, kterou podporuje dříve nainstalovaná verze Blenderu. Blender funguje i bez instalace Pythonu, ale znemožní se tím práce se skripty v Blenderu, a nejen tvorba, ale i nahrávání. Skript vytvořený mimo Blender může mít formát textového souboru, anebo Python skriptu a půjde stále spustit v Blenderu. Pokud se rozhodneme psát skripty mimo Blender, pak se do Blenderu nahrají pomocí panelu `Text Editor` na něm `Text/Open Text Block`. Skript se spustí pomocí stisknutí kláves `Alt+P`. Psaní skriptů v Blenderu je možné, ale hlášení chyb se při netriviálních chybách zobrazuje v příkazovém řádku.Pokud tedy kompilujeme přes Blender, je vhod-

né si ho spustit z příkazového řádku a sledovat vzniklé chyby zde.

```
# Modul pro vypocet souradnic kuzelu stromu

# Vlozeni potrebnych modulu
import math, random
from math import *
from random import *

# Tridy:

class kuzel:
        '''Trida kuzel graficka reprezentace vetve'''
        def __init__(self,strpods,vrchol,polpods,vyska,pouziti,ziv):
                self.S=strpods    # stred podstavy
                self.V=vrchol     # vrchol kuzele
                self.r=polpods    # polomer podstavy
                self.v=vyska      # vyska kuzele
                self.po=pouziti # jestli jiz vetvi byl prirazen stred podstavy
                self.zi=ziv      #ziva nebo mrtva vetev

class znak:
        def __init__(self,vek,term,pocBodx,pocBody,konBodx,konBody,ikuz):
                self.bAx=pocBodx
                self.bAy=pocBody
                self.bBx=konBodx
                self.bBy=konBody
                self.vek=vek
                self.term=term
                self.ikuz=ikuz
```

Obr. 3.2: Ukázka programovacího prostředí Python

3.2 Algoritmus pro tvorbu celého stromu

V této podkapitole uvedeme implementaci vygenerovaného L-systému do modelu smrku a vizualizaci celkového stromu. Uvedeme i případy, ve kterých se kvůli programování musel upravit L-systém, a jiné postupy potřebné k realizaci modelu. Algoritmus se skládá ze čtyř částí: vygenerování parametrů, dopočítání parametrů, výpočetní část pro realizaci L-systému a vizualizace.

3.2.1 Vygenerování a výpočet parametrů

Jak jsme již uvedli v podkapitole 2.3, parametry se generují jako výběr z náhodného intervalu, který jsme vhodně zvolili z poskytnutých dat. Data se vybírala pomocí random funkce rovnoměrného rozdělení. Většina parametrů je normovaná k výšce

stromu, aby se odstranily závislosti na tomto parametru při volbě z rozsahu. Algoritmus na začátku vygeneruje parametry stromu v následujícím pořadí:

- výška celého stromu,
- normovaná výška živé části koruny,
- normovaná výška mrtvé části koruny,
- normovaný průměr kmene u země,
- počet živých větví na jeden metr násobený výškou živé části koruny, čímž dostaneme celkový počet větví živé části koruny,
- hlavní poloosa půdorysné elipsy,
- vedlejší poloosa půdorysné elipsy,
- parametry p, q pro výpočet úhlu větve.

Po náhodném vygenerování parametrů stromu je potřeba ještě dopočítat a zadat některé parametry:

- Počet pater živé části, na základě počtu větví na jedno patro a celkového počtu větví, tento parametr se pouze dopočetl.
- Vzdálenost mezi patry, na základě počtu pater živé části a výšky živé části koruny, tento parametr se bere i pro mrtvou část koruny.
- Délka ročního přírůstku na větvi, tento parametr je pevně daný.

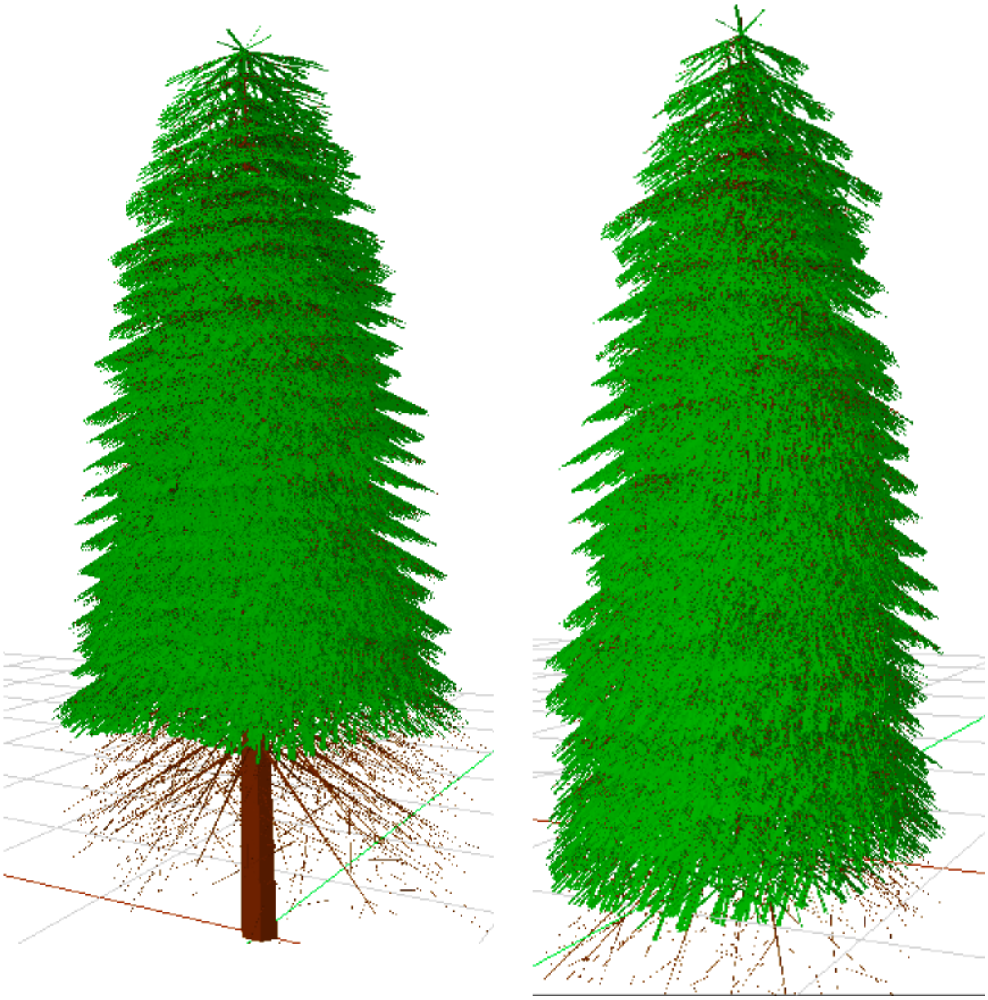
Dále je potřeba vygenerovat větve. Větve jsou sice generovány pomocí L-systémů, ale pro algoritmus jsme zavedli třídu větev, do které se vloží parametry potřebné pro volbu L-systému na této větvi a pro následnou vizualizaci. Každá třída je určena výškou, ze které vyrůstá hlavní větev z kmene a úhlem otočení ψ kolem hlavní osy kmene stromu. Parametry, jež vstupují do třídy větve jsou: délka ročního přírůstku na větvi, maximální délka větve, výška, ve které větev vyrůstá z kmene, úhel větve ϑ, úhel natočení od osy x ψ v půdorysné elipse a parametr určující, zda je větev mrtvá či živá.

Před vygenerováním třídy větve musíme ještě nechat spočítat maximální délku větve, výšku, ve které větev vyrůstá z kmene, úhel větve ϑ, úhel natočení od osy x ψ v půdorysné elipse a určit, zda je větev živá či mrtvá. Výšku, ve které vyrůstá větev z kmene, určíme pomocí parametru vzdálenost mezi patry. První patro živé části koruny vyrůstá na hranici mrtvé a živé části koruny. Dál do živé i mrtvé koruny jsou od sebe jednotlivá patra vzdálena o vzdálenost mezi patry až do začátku mrtvé koruny a do špičky stromu.

Úhel ψ se vygeneruje pomocí počtu větví na patro. Rozsah jsme zvolili o jedničku větší a menší než průměrný počet větví na patro. Parametr se generuje pro každé pa-

tro zvlášť. Když máme počet větví, už snadno dopočítáme úhel mezi každými dvěma větvemi. Větve jsou po obvodu celého kmene rozmístěny rovnoměrně s malou odchylkou. Postup se skládá z vypočítání úhlu bez odchylky a následném vygenerování odchylky pro každou větev. Tzn. $\psi = \frac{360}{p} + d_\psi$, kde p je počet větví na patro a d_ψ je přirozená odchylka. Odchylka je použita pro větší podobu reálnému smrku.

Parametry maximální délka větve a úhel větve ϑ jsme si popsali jak spočítat v podkapitole 2.3. Hodnoty potřebné pro výpočet těchto dvou parametrů jsou již vygenerovány. Hned poté, co je třída vytvořena, se do lokálních proměnných třídy přiřadí již vygenerované parametry a dopočítají se parametry α, β a γ. Jak se tyto parametry počítají, jsme si již uvedli v podkapitole 2.3. Parametry života větve přiřadíme podle toho, v jaké části koruny se větev nachází.

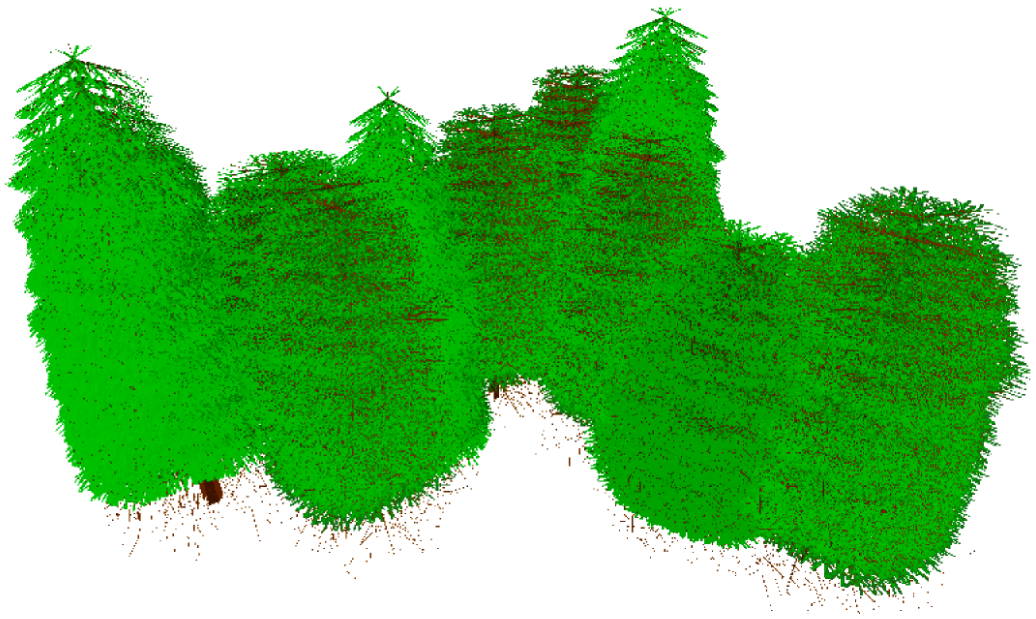


Obr. 3.3: Vymodelované stromy

3.2.2 Výpočetní část pro realizaci L-systému

Realizace L-systému se nachází ve třídě L-systému jako metoda. V rámci této metody je i vizualizace, ale o té zde mluvit nebudeme. Jelikož jsme si L-systém již popsali i je znám proces iterací, uvedeme si zde jen postupy odlišné od klasického fungování L-systémů.

Rozdílný je přístup k mocnostem pravidel, jelikož řešení přes spojité funkce by bylo na programování příliš dlouhé a zbytečně složité. Řešení je obdobné, ale pomocí modifikování podintervalů pro výběr pravidla. Jakýkoliv náhodný výběr je zde realizovaný vygenerováním náhodného čísla pomocí random funkce rovnoměrného rozdělení z intervalu $\langle 0, 1\rangle$ a ten je rozdělený na podintervaly podle toho, kolik voleb se tímto rozhodováním má volit. Následně podle toho, do jakého podintervalu padne se vybere příslušná volba (např. přepisovací pravidlo). Další úprava z pohledu časové úspory kódu je generování přepisovacího pravidla p_1. Místo toho, aby se větev postupně prodlužovala dokud nedosáhne do vzdálenosti α, se větev vytvoří rovnou v této délce. Dále se již aplikuje větvení pole pravidel p_2 až p_3.



Obr. 3.4: Les ze stromů modelovaných pomocí L-systému

Modifikování intervalu se týká pouze přepisovacích pravidel p_2 a p_4. U ostatních pravidel to není nutné. Rozhodovací interval pro výběr náhodného čísla rozdělíme

na čtyři podintervaly – a to podle možného počtu přírůstků během jednoho aplikování přepisovacího pravidla. Pomocí konstant k_1 a k_2 modifikujeme jejich velikost podle toho, v jaké vzdálenosti od kmene stromu se nacházíme. Podinterval pro přepisovací pravidlo, jež nepřidává žádné přírůstky, bude mít ve všech případech stejnou velikost. Zbylé intervaly budou svoji velikost měnit. Konstanty k_1 a k_2 zavedeme následovně:

$$k_1 = \begin{cases} 9 & \text{pro } d \in \left\langle \alpha, \frac{\beta+2\alpha}{3} \right\rangle \vee \left\langle \gamma, \frac{d_{max}+2\gamma}{3} \right\rangle, \\ 1 & \text{jinak,} \end{cases}$$

$$k_2 = \begin{cases} 9 & \text{pro } d \in \left\langle \frac{\beta+2\alpha}{3}, \frac{2\beta+\alpha}{3} \right\rangle \vee \left\langle \frac{d_{max}+2\gamma}{3}, \frac{2d_{max}+\gamma}{3} \right\rangle, \\ 1 & \text{jinak,} \end{cases}$$

kde d je vzdálenost od kmene stromu. Podintervaly jsou pak definovány pro p_2:

podinterval žádný přírůstek: $\left\langle 0, \frac{1}{12} \right\rangle$,

podinterval pro přírůstek jedné větve:$\left\langle \frac{1}{12}, \frac{k_1+1}{12} \right\rangle$,

podinterval pro přírůstek dvou větví:$\left\langle \frac{k_1+1}{12}, \frac{k_1*k_2+2}{12} \right\rangle$,

podinterval pro přírůstek tří větví:$\left\langle \frac{k_1*k_2+2}{12}, 1 \right\rangle$,

pro p_4:

podinterval žádný přírůstek: $\left\langle \frac{11}{12}, 1 \right\rangle$,

podinterval pro přírůstek jedné větve:$\left\langle \frac{k_1k_2+1}{12}, \frac{11}{12} \right\rangle$,

podinterval pro přírůstek dvou větví:$\left\langle \frac{k_1}{12}, \frac{k_1k_2+1}{12} \right\rangle$,

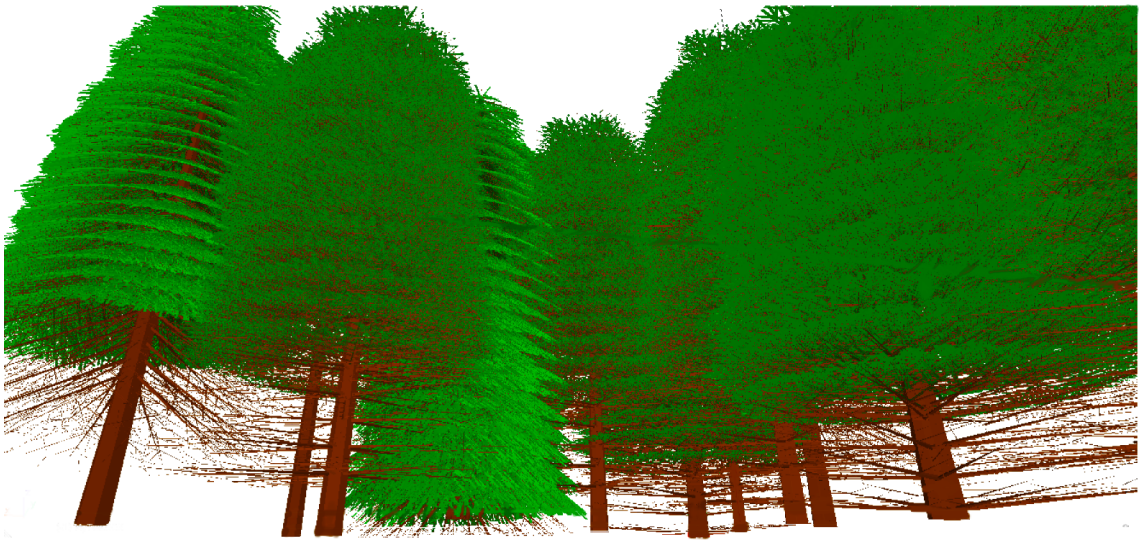
podinterval pro přírůstek tří větví:$\left\langle 0, \frac{k_1}{12} \right\rangle$.

3.2.3 Vizualizace

Vizualizace je součástí metody třídy větev. Když je tato metoda zavolána, tak se vytvoří L-systém resp. pole znaků reprezentující výsledný L-systém a po té se začne vizualizovat. Celý strom se po vizualizaci skládá z jednotlivých kuželů jako větví a ploch kolem větví jako jehličí. Kvůli vizualizaci jsme zaváděli parametr i v L-sys-

tému. Pro větev byla vytvořena samostatná třída s těmito parametry: střed podstavy kužele, vrchol kužele, poloměr podstavy kužele a výška kužele. Kužely se vkládají do pole kuželů, které se pak předá jako výstup modulu.

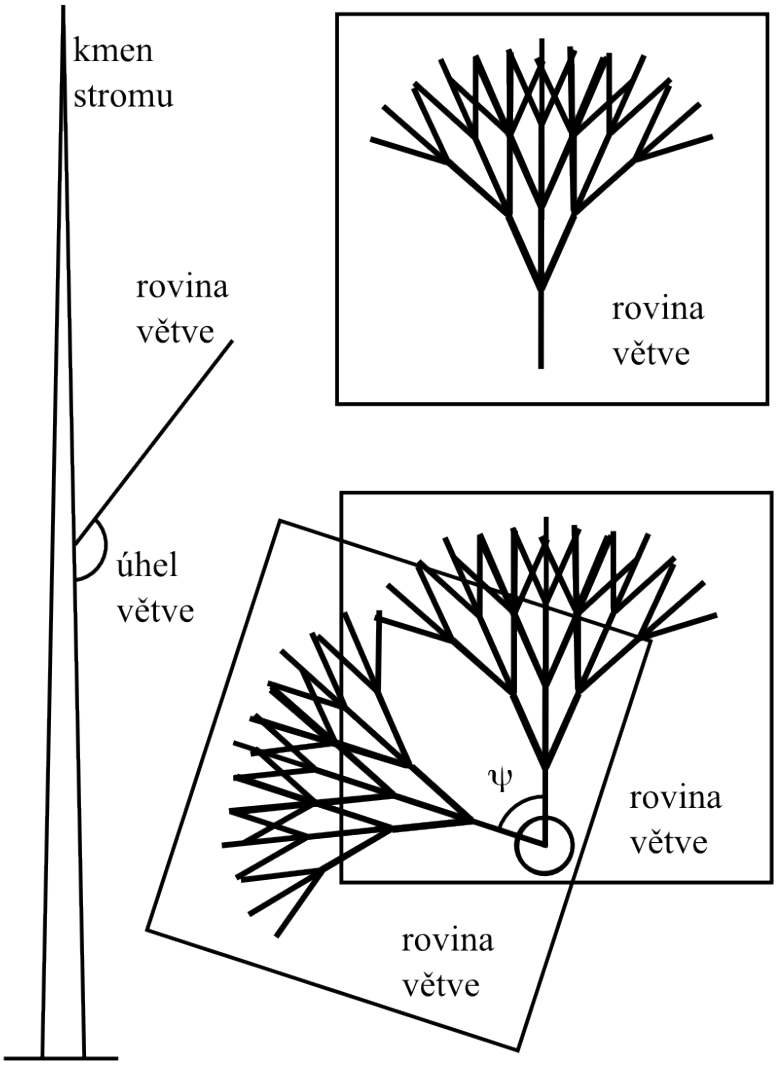
Po dokončení tvorby L-systému se vygeneruje pole kuželů o velikosti proměnné i_m podle podkapitoly 2.4. Parametr i se převáděl stejný do větví, které vyrůstaly v prodloužení původní větve, a navyšoval se nad aktuální hodnotu i_m, pokud větev rostla do kladného nebo záporného směru. Proměnná i_m se samozřejmě navýšila vždy, když jí některý parametr i převýšil. Takže poslední hodnota i_m odpovídá počtu kuželů na tomto L-systému.



Obr. 3.5: Les ze stromů modelovaných pomocí L-systému

Tvorba kuželů spočívá v procházení řetězce L-systému. Když se dojde ke znaku V nebo v, vytvoří se nebo se modifikují parametry kužele s indexem odpovídajícím parametru i aktuálního znaku. V této části také umisťujeme větev do souřadného systému stromu. Tudíž střed a vrchol kužele jsou na rozdíl od parametrů L-systému určeny třísložkovým polohovým vektorem. Transformace, o nichž jsme se zmínili v podkapitole 2.4, se provedou zde – viz obr. 3.6. Vždy ověříme, zda kužel s indexem rovným parametru i aktuálního znaku již existuje nebo ho budeme teprve vytvářet. Pokud již existuje, pak jen dojde k prodloužení kužele o délku ročního přírůstku. Pokud kužel ještě neexistuje, pak se vytvoří všechny parametry kužele, jež jsme zmínili výše. Parametry se vytvoří přetransformováním parametrů A a B do souřadné soustavy stromu a dopočítají se zbylé parametry. Výška kužele se dopočítá jen přidáním délky ročního přírůstku k stávající délce kužele a poloměr

kužele se vypočítá z výšky kužele. Platí, že $v = 200r$, kde v je výška kužele a r poloměr.



Obr. 3.6: Transformace větve do prostoru

Když se v každé třídě větve vytvoří pole kuželů, přidá se do pole kuželů celkového stromu. Tímto jsme se přesunuli do tvorby skriptů přímo pracujících s grafickými knihovnami Blenderu. Blender pracuje s voxely a facemi. Pomocí těchto dvou prvků vytvoříme kužely, které máme v poli kuželů stromu. Pro tvorbu kuželů potřebujeme jen znát poloměr podstavy, střed podstavy a vrchol kužele. Každý kužel je vytvořen jako mesh objekt, který má mnoho vlastností. Vlastnosti, které využijeme jsou materiál, barva a průhlednost. Materiál a barva jsou jasné z hlediska vzhledu stromu.

Průhlednost potřebujeme na tvorbu jehlic. Jehlice nemůžeme tvořit po jedné, protože by to nebylo časově realizovatelné a objekt by pak měl obrovské paměťové nároky na uložení. Jehlice proto vytvoříme jako plochy kolem kužele v jedné rovině. Plochy, ale musí mít nějakou průhlednost, protože pak by model nebyl reálný a plochy jehlic by vypadaly jako plné listy a ne jednotlivé jehlice. Vytvořené modely stromů jsou vidět na obrázcích 3.3, 3.4 a 3.5.

ZÁVĚR

V rámci diplomové práce byl vytvořen L-systém, který vizualizuje jehličnatý strom, konkrétně smrk. L-sytém sice nepostihuje tvorbu celého stromu, ale je pomocí něj vytvořena nejkomplexnější část stromu. Jelikož cílem bylo vytvoření algoritmu na tvorbu celého stromu, je v práci uveden kromě postupu tvorby L-systému také zbytek algoritmu. Jsou zde uvedené i postupy a výpočty některých parametrů.

Práce řeší reálnou úlohu a předpokládá se její praktické využití. Jelikož zadání bylo podmíněno originalitou každého vygenerovaného modelu a reálným vzhledem, byla volba L-systémů nejvhodnější. L-systém je schopen držet se základních charakteristik modelu, ale v zásadě vnést náhodné prvky do celého modelu. L-systém by se dal i více přiblížit reálnému stromu, ale požadavky na využití modelu to nevyžadovaly. Algoritmus na modelování smrku byl vytvořen pro využití na generování jednotlivých stromů do modelu lesa. Hlavní na modelu stromu je hustota zeleně. Díky L-systému stačí změnit pouze hodnoty parametrů α, β nebo γ a vygenerovat nové stromy.

Návaznost na tuto práci může být tvorba modelů pro další druhy stromů. Díky obecným vlastnostem L-systémů nebude složité upravit L-systém pro tvorbu jiného druhu stromu. Nejjednodušší bude modifikace pro jiné jehličnaté stromy, ale ani pro listnaté stromy nebude algoritmus zcela jiný. V případě listnatých stromů by L-systém tvořil již celý strom ne jen jednotlivé větve jako u jehličnatých stromů. L-systémy by samozřejmě mohly tvořit i celý jehličnatý strom, ale kvůli časovým a paměťovým nárokům jsme tuto volbu zamítli.

ZÁKLADNÍ POJMY A ZNAČENÍ

Uvedeme seznam pojmů a používané značení:

Σ	$\cdots$	konečná abeceda
$\alpha, \beta, \ldots$	$\cdots$	slova nad abecedou Σ (pouze v kapitole 1)
$\lvert\alpha\rvert$	$\cdots$	délka slova α
$\#_a(\alpha)$	$\cdots$	četnost symbolu a ve slově α
ε	$\cdots$	prázdné slovo abecedy Σ
Σ^+	$\cdots$	množina všech konečných neprázdných slov nad abecedou Σ
Σ^*	$\cdots$	$\Sigma^* = \Sigma^+ \cup \{e\}$
$\alpha \cdot \beta$	$\cdots$	zřetězení slov, $\alpha \cdot \beta = \alpha\beta$
α	$\cdots$	je podslovo slova β, pokud platí $\beta = lc\alpha rc$, $lc, rc \in \Sigma^*$
L	$\cdots$	jazyk nad abecedou Σ, $L \subseteq \Sigma^*$
(lc, rc)	$\cdots$	kontext nad abecedou Σ, $lc\alpha rc \in \Sigma^*$
lc	$\cdots$	levý kontext, $lc \in \Sigma^*$
rc	$\cdots$	pravý kontext, $rc \in \Sigma^*$
T	$\cdots$	množina terminálů,
N	$\cdots$	množina neterminálů, $T \cup N = \Sigma \wedge T \cap N = \emptyset$
$\mathcal{G}$	$\cdots$	gramatika $\mathcal{G} = (N, T, P, \sigma)$
σ	$\cdots$	axiom $\sigma \in \Sigma^+$
P	$\cdots$	množina přepisovacích pravidel $P \subseteq \Sigma^* \cdot N \cdot \Sigma^* \times \Sigma^*$
$\mathcal{L}$	$\cdots$	L-systém $\mathcal{L} = (\Sigma, P, \sigma)$
IL-systém	$\cdots$	kontextový L-systém
0L-systém	$\cdots$	bezkontextový L-systém
DL-systém	$\cdots$	deterministický L-systém
EL-systém	$\cdots$	rozšířený L-systém
parametrický L-systém	$\cdots$	parametrický L-systém
nedeterministický L-systém	$\cdots$	nedeterministický (stochastický) L-systém

SEZNAM OBRÁZKŮ

OBSAH PŘILOŽENÉHO CD

- Python skript – generování parametrů, tvorba L-systémů, výpočty, vizualizace (tvorba kuželů),
- textový soubor – skript implementace kuželů vytvořených předešlým skriptem do Blenderu, nutno importovat do Blenderu, samostatně nefunkční
- ukázky vytvořených 3D modelů.

LITERATURA

[1] JANOUTOVÁ R. *Matematické modelování pomocí L-systémů*. Brno: Vysoké učení technické v Brně, Fakulta strojního inženýrství, 2010. 30 s. Vedoucí bakalářské práce doc. PaeDr. Dalibor Martišek, Ph.D.

[2] KOLKA M. *L-systems: new results and application*. Brno: BUT Brno, 2004.

[3] ŽÁRA a kol. *Moderní počítačová grafika*. Praha: Grada, 1998.

[4] PRUSINKIEWICZ P., LINDENMAYER A. *The Algorithmic Beauty of Plants*. New York: Springer-Verlag, 1990, dotisk v 1996.

[5] MULLEN T. *Mastering Blender*. Indianopolis, Indiana: Wiley Publishing, Inc. 2009.

[6] ŠVEC J. *Učebnice jazyka Python (aneb Létající cirkus)*[online]. 2002. Dostupné z: <http://www.py.cz/UcebniceJazykaPython>.

[7] PILGRIM M. *Ponořme se do Python(u) 3*. Praha: CZ.NIC, z. s. p. o. 2010.

[8] DOLANSKÝ T., HOMOLOVÁ L., HANUŠ J. *Zpracování laserových dat smrkového porostu v prostředí ArcGIS*[poster]. 17. konference GIS ESRI v ČR. Praha, 2008.