FAKULTA MECHATRONIKY,
INFORMATIKY A MEZIOBOROVÝCH
STUDIÍ TUL

Diplomová práce

Rozpoznávání tloušťky nanovláken: vizualizace vláken imitující data ze skenovací elektronové mikroskopie

Studijní program: N0613A140028 Informační technologie
Autor práce: **Bc. Ondřej Mach**
Vedoucí práce: Ing. Pavel Márton, Ph.D.
Ústav mechatroniky a technické informatiky

Liberec 2024

FAKULTA MECHATRONIKY,
INFORMATIKY A MEZIOBOROVÝCH
STUDIÍ TUL

Zadání diplomové práce

Rozpoznávání tloušťky nanovláken: vizualizace vláken imitující data ze skenovací elektronové mikroskopie

Jméno a příjmení:	**Bc. Ondřej Mach**
Osobní číslo:	M22000025
Studijní program:	N0613A140028 Informační technologie
Zadávající katedra:	Ústav mechatroniky a technické informatiky
Akademický rok:	2023/2024

Zásady pro vypracování:

Diplomová práce bude příspěvkem k vytvoření trénovacích dat pro strojové rozpoznávání tloušťky nanovláken na základě dat ze skenovací elektronové mikroskopie (SEM).

1. Proveďte rešerši principu SEM a přístrojů využívaných na KCH FM TUL pro vizualizaci nanovláken.
2. Ve spolupráci s D. Šafaříkem specifikujte datový formát, ve kterém budou kompletní informace o charakteru modelovaných vláken předány k vizualizaci.
3. Vytvořte softwarový nástroj pro vizualizaci počítačově generovaných sad nanovláken se zohledněním získaných informací o metodě SEM. Pokuste se o co nejlepší věrnost a podobnost reálným datům. Parametrizujte vizualizaci tak, aby bylo možné zohlednit různé povrchové charakteristiky (např. hladkost) nanovláken, kvalitu a plošnou distribuci pokovení nanovláken atd. Nástroj musí být dimenzován k vizualizaci velkého množství trénovacích dat. Výstupem budou obrázky imitující SEM, typicky 1024×1024, png, v odstínech šedi, a datové soubory/obrázky obsahující klíčové vlastnosti nanovláken (tloušťku, orientaci atd.) na stejném půdorysu.
4. Vizualizujte alespoň jednu deformaci/odlišnost nanovlákna od typického tvaru, která je charakteristická pro některý režim elektrostatického zvlákňování/zvlákňovaný materiál.
5. Sledujte a diskutujte vliv zahrnutí jednotlivých aspektů trénovacích dat na kvalitu rozpoznávání tloušťky vláken provedené s využítím externí neuronové sítě na reálných datech ze SEM.
6. Vytvořený zdrojový kód důkladně okomentujte.
7. Sepište diplomovou práci v požadovaném rozsahu.

Rozsah grafických prací:	dle potřeby dokumentace
Rozsah pracovní zprávy:	40 až 50 stran
Forma zpracování práce:	tištěná/elektronická
Jazyk práce:	čeština

Seznam odborné literatury:

[1] David Lukáš: Nanovlákna: teorie, technologie a použití, Academia 2023.
[2] J. Žára, B. Beneš, P. Felkel: Moderní počítačová grafika, Computer press 1998.
[3] O. Ronneberger, P. Fischer, T. Brox: U-Net: Convolutional Networks for Biomedical Image Segmentation, arXiv:1505.04597v1 2015.

Vedoucí práce:	Ing. Pavel Márton, Ph.D. Ústav mechatroniky a technické informatiky
Datum zadání práce:	12. října 2023
Předpokládaný termín odevzdání:	14. května 2024

L.S.

prof. Ing. Zdeněk Plíva, Ph.D.
děkan

doc. RNDr. Pavel Satrapa, Ph.D.
garant studijního programu

V Liberci dne 12. října 2023

Prohlášení

Prohlašuji, že svou diplomovou práci jsem vypracoval samostatně jako původní dílo s použitím uvedené literatury a na základě konzultací s vedoucím mé diplomové práce a konzultantem.

Jsem si vědom toho, že na mou diplomovou práci se plně vztahuje zákon č. 121/2000 Sb., o právu autorském, zejména § 60 – školní dílo.

Beru na vědomí, že Technická univerzita v Liberci nezasahuje do mých autorských práv užitím mé diplomové práce pro vnitřní potřebu Technické univerzity v Liberci.

Užiji-li diplomovou práci nebo poskytnu-li licenci k jejímu využití, jsem si vědom povinnosti informovat o této skutečnosti Technickou univerzitu v Liberci; v tomto případě má Technická univerzita v Liberci právo ode mne požadovat úhradu nákladů, které vynaložila na vytvoření díla, až do jejich skutečné výše.

Současně čestně prohlašuji, že text elektronické podoby práce vložený do IS/STAG se shoduje s textem tištěné podoby práce.

Beru na vědomí, že má diplomová práce bude zveřejněna Technickou univerzitou v Liberci v souladu s § 47b zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů.

Jsem si vědom následků, které podle zákona o vysokých školách mohou vyplývat z porušení tohoto prohlášení.

13. května 2024 Bc. Ondřej Mach

Rozpoznávání tloušťky nanovláken: vizualizace vláken imitující data ze skenovací elektronové mikroskopie

Abstrakt

Tato diplomová práce se věnuje tvorbě softwaru pro generování syntetických snímků nanovláken o známých rozměrech a dalších informacích. Snímky mají za cíl simulovat reálné výstupy ze skenovacího elektronového mikroskopu (SEM). Tato data by následně měla být použita pro trénování neuronové sítě s cílem zisku nástroje pro určení průměru reálných nanovláken pro KCH TUL, především pro oddělení Bioinženýrství. Za tímto účelem bylo zvoleno použití jazyku Python a nástroje Blender. Výsledný skript má několik způsobů nastavení a poskytuje uživateli možnost generovat snímky s vlákny o různých parametrech. Závěrečným krokem bylo také testování, jak si několik generovaných sad povede. Navržený software poskytuje základ pro další rozvoj metod automatického rozpoznávání snímků nanovláken.

Klíčová slova

Blender, nanovlákna, neuronové sítě, Python, SEM

Nanofiber-thickness recognition: fiber visualization mimicking scanning electron microscopy data

Abstract

This master's thesis focuses on creating software for generating synthetic images of nanofibers with known dimensions and other information. The images aim to simulate real outputs from a scanning electron microscope (SEM). This data should subsequently be used to train a neural network with the goal of developing a tool for determining the diameter of real nanofibers for the KCH TUL, particularly the Department of Bioengineering. For this purpose, Python language and Blender tool were chosen. The resulting script the resulting script has several ways to set it up and provides the user with the ability to generate images with fibers of various parameters. Additionally, testing was conducted to evaluate the performance of several generated datasets. The proposed software provides a foundation for further development of methods for automatic recognition of nanofiber images.

Keywords

Blender, nanofibers, neural networks, Python, SEM

Poděkování

V první řadě bych rád poděkoval vedoucímu práce, panu doktoru Pavlu Mártonovi, za jeho cenné rady a vedení celého projektu. Trpělivost během mého výzkumu a psaní této diplomové práce. Dále bych chtěl poděkovat svému kolegovi Davidu Šafaříkovi za jeho spolupráci a podíl na projektu.

Rovněž bych rád vyjádřil velké díky celému oddělení Bioinženýrství TUL za jejich konzultace a přínosy k mé práci. Zejména chci poděkovat konzultantovi práce panu inženýru Jaroslavu Mikule a také paní docentce Evě Kuželové Košťákové.

Nemohu zapomenout na svoji rodinu, zejména na mého tatínka, který mě podporoval nejen ve smyslu zázemí, ale také finančně. A to po celou dobu mého studia. Bez něj bych nemohl studovat. Děkuji také mamince, za její neustálou podporu a za pomoc při korekci práce.

Na závěr bych rád vzdal speciální dík mé bývalé spolužačce Emě Chudobové za její trpělivost s mými někdy hloupými dotazy a ochotu pomoci.

Děkuji Všem za jejich podporu a přínos, ať už přímý nebo nepřímý, k této práci. Bez nikoho ze zmíněných by práce nemohla dosáhnout finální podoby.

Obsah

Seznam obrázků

Seznam tabulek

Seznam zdrojových kódů

Seznam zkratek

AC	Alternating Current, střídavý proud
API	Application Programming Interface
BOR	borders, okraje
BSDF	Bidirectional Scattering Distribution Function
DC	Direct Current, stejnosměrný proud
EEVEE	Extra Easy Environment Engine
FM	Fakulta mechatroniky, informatiky a mezioborových studií
FP	Fakulta přírodovědně-humanitní a pedagogická
FXAA	Fast Approximate Anti-Aliasing
HSV	Hue Saturation Value
IOR	Index of Reflection, index odrazu
KCH	Katedra chemie
NumPy	Numerical Python
ORI	orientation, orientace
PIL	Python Imaging Library
RGB	Red Green Blue
SciPy	Scientific Python
SEG	segmentation, segmentace
SEM	skenovací elektronový mikroskop
sRGB	Standard Red Green Blue
THI	thickness, tloušťka
TTL	Through-the-Lens
TUL	Technická univerzita v Liberci
ReLU	Rectified Linear Unit
UV	ultrafialový
VTK	Visualization Toolkit
2D	dvourozměrný
3D	trojrozměrný

1 Úvod

Nanomateriály jsou jednou z oblastí vědeckého výzkumu a technologického vývoje, která v moderní době zaznamenala nemalý rozvoj, a to hlavně svým ohromným potenciálem pro různé aplikace v průmyslu, medicíně, elektronice a mnoha dalších odvětvích. Vyznačují se unikátními vlastnostmi, které jsou způsobeny jejich extrémně malými rozměry a specifickou strukturou na úrovni nanometrů. V našem rychle se rozvíjejícím světě přináší nanomateriály a nanovlákna inovativní řešení pro mnoho současných problémů a otevírají dveře k novým možnostem v oblasti materiálového designu a technologie.

Věda nanomateriálů a jejich zkoumání s sebou přinesla také spojení jiných vědeckých disciplín a oblastí technologie. Nejen poznatky z fyziky, chemie, biologie, materiálové vědy a inženýrství, jež jsou její nedílnou součástí, ale také jiné vědní obory přispívají k jejímu vývoji Mezi ně lze řadit například zpracování dat v rámci oboru informačních technologií. Tato symbióza různých oborů posiluje výzkumné úsilí a přispívá k objevům, které by jinak nebyly možné. Výsledkem je neustálý pokrok v oblasti nanomateriálů a nanovláken, který ovlivňuje mnoho aspektů našeho každodenního života a otevírá cestu k novým technologickým inovacím a aplikacím.

Význam nanomateriálů spočívá v unikátních vlastnostech vyplývajících z jejich extrémně malého rozměru a specifické struktury. Tyto materiály mohou mít odlišné chemické, fyzikální a mechanické vlastnosti oproti jejich makroskopickým protějškům.

Pro zkoumání nanomateriálů jsou využívány speciální přístroje, které umožňují pozorování a analýzu jejich struktury a vlastností na mikroskopické úrovni. Jedním z nejvíce využívaných přístrojů pro zkoumání nanomateriálů je skenovací elektronový mikroskop (SEM). SEM je schopen poskytnout obrazové informace o povrchové struktuře vzorku s vysokým rozlišením pomocí zvýšeného zobrazení povrchových detailů prostřednictvím elektronového paprsku. Na základě snímků SEM je možné určovat i tloušťku jednotlivých vláken nanomateriálu. Tento přístroj je nepostradatelný pro studium morfologie a struktury nanomateriálů a poskytuje důležité informace pro další analýzy a výzkum. [1, 2]

Tato práce bude příspěvkem k vytvořením trénovacích dat pro strojové rozpoznávání tloušťky nanovláken na základě dat ze SEM. To by mělo umožnit vytvoření softwaru pro usnadnění práce při měření průměru nanovláken a nahradit stávající proces tohoto měření. Práce si klade za úkol rešerši principu SEM a zkoumání vlivu různorodých faktorů na konečnou podobu snímku. Využití získaných informací při vizualizaci v rámci snímků imitujících realitu, společně s využitím poznatků z počítačové grafiky.

2 Teoretická část

2.1 Nanovlákna a jejich tvorba

Definice nanovláken není zcela jednoznačná. Společně s dalšími nanomateriály se však od klasických materiálů vyznačují změnou fyzikálních vlastností, která je způsobená jejich rozměrem. Pro účely této práce je definujme stejným způsobem jako komunita věnující se jejich problematice, a to pomocí minimálního délkového poměru (aspect ratio). Tedy dle Wilsona [3]: „Nanovlákna jsou tradičně definována jako válcové struktury s vnějším průměrem pod 1000 nm a délkovým poměrem – poměrem mezi délkou a šířkou – větším než 50."

Pro tuto práci jsou hlavní oblastí zájmu polymerní nanovlákna. Ta jsou připravována výrazně odlišnými způsoby a postupu než klasické mikrovlákenné textilní materiály. Základem pro polymerní nanovlákna je rozpouštědlo (případně kombinace více rozpouštědel) a směs polymerního materiálu. Tato směs se dávkuje využitím trysek a podstatná je především její „zvláknitelnost". Příkladem zvláknitelných látek mohou být například fosfolipidy. [1, 4]

Jak již bylo řečeno, způsoby výroby nanomateriálů jsou od klasických postupů odlišné. Tyto technologie výroby jsou různé a využívají také různých sil, které formují vlákno materiálu. Pro přehlednost je uvedena převzatá Tabulka 2.1.

Tabulka 2.1: Přehled technologií výroby polymerních nanovláken [1]

Technologie výroby polymerních nanovláken z polymerních kapalin (česky/anglicky)		**Přetvářecí síla formující vlákno**
elektrické zvlákňování	electrospinning	elektrická síla
rozfukování taveniny polymeru	meltblown	hydrodynamická síla tření taveniny o vzduch
rozfukování roztoku polymeru	air-jet spinning / solution blowing	hydrodynamická síla tření roztoku o vzduch
odstředivé zvlákňování	centrifugal spinning / rotary jet spinning / forcespinning	mechanická odstředivá síla
tažení vláken	drawing	mechanická síla
výroba bikomponentních vláken typu „Ostrovy v moři"	islands-in-the-sea bicomponent fibers	mechanická síla
expanzní zvlákňování	flash spinning	hydrodynamická síla a termodynamické jevy
laserové zvlákňování	laser spinning	hydrodynamická síla
smykové zvlákňování	shear spinning	hydrodynamická a smyková síla

Ze zmíněných technologií si dovolím primárně zaměřit na metodu electrospinning. Především pro jeho hojné využívání na KCH TUL, a také protože takto produkované materiály jsou primárním zájmem této práce.

Metoda electrospinning využívá elektrickou sílu, a jedná se tedy o elektrické zvlákňování. Elektrické zvlákňování využívá potenciálový rozdíl k dosažení nadkritické hodnoty intenzity elektrického pole na povrchu kapaliny, typicky polymerického roztoku. Tím vzniká lokální zvýšení elektrického tlaku nad kapilárním tlakem, vyvolaným povrchovým napětím a křivostí kapalinového tělesa. Tato situace vyústí v hydrodynamickou nestabilitu, kdy se polymerní trysky transformují odpařováním rozpouštědla (nebo ochlazením taveniny) na submikronová vlákna nebo nanovlákna. [1]

Metodu electrospinning lze dále dělit dle konkrétních technologických postupů. Tyto postupy se odlišují na základě rozličných prvků přípravy vláken. Zásadním prvkem, který je pozměňován pro jednotlivé technologie, je polymerní roztok a jeho forma (lze např. užít pěnu nebo formu bublin). Dále je užíváno různých druhů rozpouštědel a podmínek jeho použití (např. subkritické podmínky). Rozpouštědlo je možné také nepoužít a zvlákňovat užitím UV zářením za určitých podmínek, užívání různých typů trysek i jehel a používání aktivního, či neaktivního kolektoru, což je elektroda zachycující vlákna. A nakonec s tím související užívání stejnosměrného, či střídavého proudu. [1, 5–7]

Právě užití stejnosměrného a střídavého proudu přináší dvě přední technologie, a to DC electrospinning a AC electrospinning. DC electrospinning je aktuálně nejrozšířenější metoda pro výrobu nanovláken. Využívá dva typy elektrod – zvlákňovací, od té kapalina vstupuje do zvlákňovacího prostoru, a kolektor, který má za úkol sběr vzniklých nanovláken. Mezi elektrodami je rozdíl elektrických potenciálů. Polymerní materiál se pak přivádí nejčastěji kapilárou. Úlohu kapiláry může plnit jehla, či injekční stříkačka. Při připojení zdrojů vysokého napětí nebo uzemnění k zvlákňovací elektrodě a kolektoru ve zvlákňovacím prostoru s vhodně zvoleným rozdílem potenciálů se vytvoří tzv. polymerní trysky. To umožňuje transformaci polymerní kapaliny na nanovlákna. Samotný electrospinning lze dělit dle užité zvlákňovací elektrody na bezjehlový DC electrospinning a kapilární DC electrospinning (s jehlou). [1]

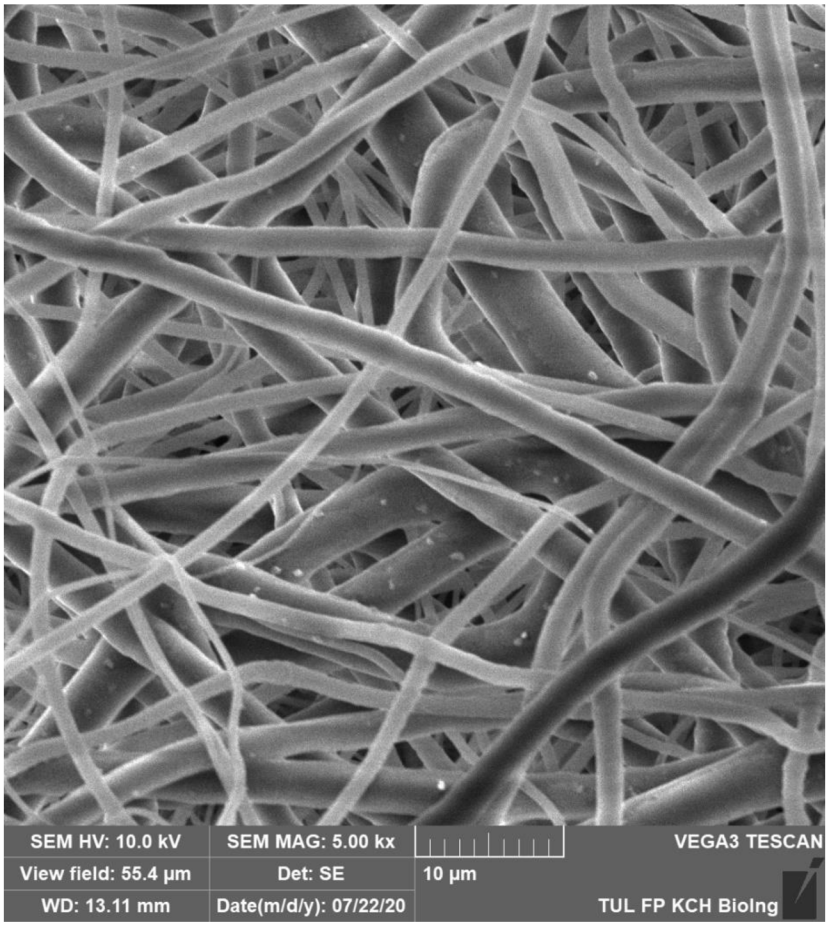
Vlákna vzniklá z electrospinningu, ale i jiných postupů, jsou ovlivňována různými parametry. Ty lze dělit na procesní a materiálové. Materiálové jsou závislé na rozpouštědle a použitém polymeru. U polymeru pak rozhoduje koncentrace, povrchové napětí, vodivost a aditivita. Procesně jsou dány nejen konstrukčními parametry procesu, ale ovlivňují ho i okolní podmínky a volba fyzikálních podmínek, při kterých proces probíhá. Po stránce procesních parametrů je důležitá nejen zvlákňovací elektroda, ale též kolektor. Kolektorů existují různé druhy a v základu se dělí na statické a dynamické. Tvar a pohyb kolektoru ovlivňuje u nově zvlákněných vláken ukládání, tvar a orientaci. Tím ovlivňuje i strukturu, kterou reflektují snímky vláken. Za zmínku stojí i podkladový materiál pro tvorbu nanovláken. Další procesní parametry, které je třeba kontrolovat, jsou velikost elektrického pole, vzdálenost mezi kolektorem a zvlákňovací elektrodou, teplota a relativní vlhkost. U materiálových parametrů se při výrobě řeší: viskozita, povrchové napětí a elektrická vodivost. Vodivost je ovlivněná koncentrací roztoku a molekulární hmotností polymeru. Povrchové napětí a elektrická vodivost také značně ovlivňují rozpouštědla. [1, 8]

Konkrétní vliv zmíněných materiálových parametrů se na struktuře vláken (a tedy i výstupních snímků) projeví následovně. Vyšší viskozita vede k vláknům s většími průměry a s typickým vláknovým vzhledem vzniklého materiálu. Nízko-viskózní tvoří kulaté útvary, spíše než vlákna. Pro povrchové napětí je to obráceně. Vláken s větším průměrem a vláknitým charakterem materiálu lze dosáhnout snižováním povrchového napětí. A nakonec elektrickou vodivostí se při vyšších hodnotách dostáváme na vlákna menší tloušťky. [1]

Uvedené parametry jsou obecnějšího charakteru. Pro specifické tvary a strukturu vláken je třeba hlubší porozumění chemické stránce věci. Výstupem jsou pak vlákna více homogenní, nebo naopak heterogenní. Vlákna hladká, porézní, s integrovanými částicemi, či s kapénkovými defekty. Vlákna s kruhovým průřezem, ale i vlákna páskovitá.

2.2 Základní princip skenovací elektronové mikroskopie (SEM)

SEM, tedy skenovací elektronovou mikroskopii (také známou jako rastrovací), lze považovat za jeden z nejužitečnějších nástrojů pro zkoumání a analýzu nanomateriálů, a tedy i nanovláken. Pomocí této metody je možné studium složení materiálu a vizualizace jeho struktury. Optické mikroskopy užívající optickou čočku dosahují okolo 200 nm, zatímco elektronový mikroskop je schopen rozlišení desítek a jednotek nanometrů, někdy až pod 1 nm. [9, 10]



Obrázek 2.1: Snímek jednoho z mnoha typů nanovláken. Poskytnutý TUL FP KCH Bioing. Snímek obsahuje také legendu s údaji o měření. Užitý materiál je polykaprolakton

Následující část bude věnovaná několika slovům SEM, jeho fungování, součástem a signálům. To vše pro lepší porozumění snímků, které tvoří jeho výstup.

Princip fungování SEM je založen na využití elektronového paprsku namísto viditelného světla, jak je tomu u klasických světelných mikroskopů. Elektrony jsou urychlovány ve vakuovém prostředí pomocí elektrického pole a zaměřovány do tenkého svazku. Vzorek musí být pro zkoumání připraven tak, aby byl elektricky vodivý. Případně použít jinou technologii (např. pomocí komory s vakuem). To obvykle zahrnuje pokrytí vzorku tenkou vrstvou kovu, jako je zlato nebo wolfram, aby se zajistila vodivost a minimalizoval se vliv náboje vzorku na výsledný obraz. Když elektronový paprsek dopadá na povrch vzorku, interaguje s atomy a generuje různé typy signálů. Tyto signály jsou zachyceny detektory v SEM, amplifikovány a přeměněny na elektrické signály. Pohyb elektronového paprsku po povrchu vzorku se řídí skenovacím vzorem, který obvykle získává obrázky bod po bodu. Intenzita zachycených signálů je mapována na odpovídající umístění na obrazovce promítající snímek, čímž vzniká obraz povrchu vzorku. Zpracování obrazu se provádí pomocí počítače, který vytváří obrazové reprezentace povrchu vzorku s vysokým rozlišením a kontrastem. Tímto způsobem SEM umožňuje detailní zkoumání mikrostruktury a morfologie povrchů různých materiálů a vzorků, což je klíčové pro mnoho vědeckých disciplín a průmyslových aplikací. [2, 9, 10]

Signály, které je možné zaznamenat, jsou: sekundární elektrony, zpětně odražené elektrony a charakteristické rentgenové záření. Pro tuto práci jsou klíčové hlavně elektronové signály. Zpětně odražené elektrony vznikají pružným rozptylem elektronů primárního paprsku s jádry atomu. Sekundární elektrony pak elektrony vyraženy z vnějších vrstev elektronové slupky atomů vzorku. Kontrast, který tvoří je topografický. [2, 9, 10]

Sběr signálu provádí několik detektorů. Everhart-Thornley detektor je schopen detekce sekundárních a zpětně odražených elektronů. Užívá se ovšem primárně pro zaznamenání sekundárních elektronových obrazů povrchu vzorku. Everhart-Thornley je detektor typu fotonásobič. Je umístěn po straně hlavní části SEM a míří šikmo na plochu se vzorkem (úhel okolo 30°). Právě díky tomuto umístění vzniká snímek, který lze připodobnit ke sledování vzorku směrem dolů, zatímco vzorek je jakoby osvětlen z boku zdrojem světla – zmíněným detektorem. Tím vzniká topografie, kde jsou rysy buď nasvíceny, nebo ve stínu, a to v závislosti na jejich relativní poloze vzhledem k detektoru. Dalším z detektorů je Through-the-Lens (TTL). TTL je uložen v hlavní sloupcové části těla mikroskopu a je schopen zachytit sekundární elektrony, které jsou emitovány z povrchu vzorku. Obraz vytvořený pomocí TTL detektoru je vysokého rozlišení, má dobrý poměr signálu k šumu a je bohatý na kontrast. Je velmi užitečný pro zobrazování důlků a prohlubní materiálu. Posledním z detektorů, které si dovolím zmínit je detektor zpětně odražených elektronů. Většina těchto elektronů je odražena směrem nahoru, a proto je tento detektor ve většině případů umístěn nad vzorkem. Snímky obsahují informaci o kontrastu složení a hustotě ve vzorku. [1, 10, 11]

Sonda elektronu prochází vzorek a zaznamenává intenzitu pro danou pozici. Zmíněné detektory fungují jako rozhraní mezi snímkem a vzorkem. Jas na snímku odpovídá síle signálu z bodu ve vzorku. Snímek proto tvoří mapu rozložení intenzity elektronových signálů odvozených ze vzorku. Během zobrazování je síla signálu úměrná počtu elektronů vyvržených z bodu na vzorku. Místa paprsku na vzorku, odkud je generován signál, se nazývají obrazové prvky s přibližně odpovídajícím počtem pixelů v obrazu zobrazeném na projekční ploše.

Analogový signál získaný z každého obrazového prvku vzorku je měřen, zesílen a převeden na napěťový signál elektronikou detektoru. Využívá se povětšinou 256 hodnot šedého tónu, které jsou uloženy formou 8 bit. Pro získané snímky je klíčový kontrast, díky němuž lze materiál analyzovat. Ten je ovlivněn několika prvky. Pro tuto práci bych si dovolil zmínit: změny topografie vzorku, rozdíly ve složení, povrchový potenciál a elektrickou vodivost. [2]

Součástí práce bylo zpracovat i další z přístrojů pro vizualizaci nanovláken na KCH TUL. Pro potřeby projektu však žádný z nástrojů neposkytoval žádnou výhodu po stránce informační, ani po stránce vizualizační. Z tohoto důvodu je část věnovaná přístrojům věnována čistě SEM.

2.3 Kontrast, světlé a tmavé oblasti snímků SEM

Na konečný vzhled snímku má vliv vícero faktorů. Jedním z typických jevů, které je možné na snímcích pozorovat, jsou vlákna v různých odstínech šedé (od bílé až po téměř černou barvu). Dále pak tmavší a světlejší oblasti. Na základě konzultací a rešerše si dovolím uvést některé z důvodů, které k těmto jevům vedou. Pro výsledný snímek je totiž klíčový kontrast – právě díky němu vznikají světlejší, či tmavší části výstupu, a je možné materiál analyzovat. Je tedy důležité popsat, co vše tento kontrast ovlivňuje, aby bylo možné přiblížit se během tvorby syntetických dat co nejvíce realitě.

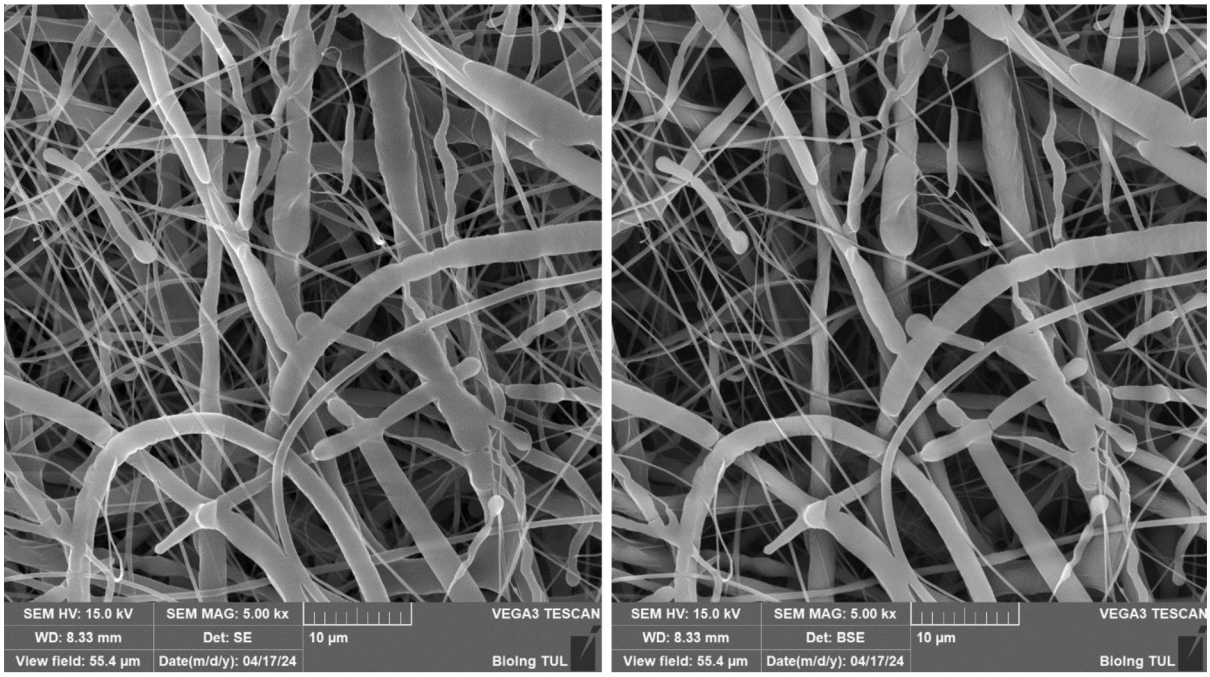
Prvním z jevů je souvislost atomového čísla s množstvím zpětně odražených elektronů. Tento jev je znám jako atomový (kompoziční) kontrast. Vzorek, respektive oblast vzorku, s vyšším atomovým číslem bude mít relativně jasnější vzhled kvůli většímu množství zpětně odražených elektronů, zatímco oblast s nižším atomovým číslem bude vypadat relativně tmavěji. Světlé oblasti tedy obsahují těžší prvky, zatímco části zdánlivě tmavší představují prvky s nižším atomovým číslem. [2, 12]

Další z kontrastů je topografický a vzniká díky sekundárním elektronům. Tyto elektrony jsou primárně zachyceny Everhart-Thornley detektorem (viz podkapitola 2.2). Vlivem nízké energie elektronů na povrchu materiálu dochází k jejich snadnějšímu opuštění z vrchních vrstev a poslouží tak k vytvoření topografického kontrastu. Následkem toho je, že oblasti směřující k detektoru jsou světlejší. Tedy generují vyšší kontrast ve srovnání s těmi, které jsou obráceny od něj. Ty jsou naopak tmavší. [2, 12]

Uvedené jevy lze dále ovlivňovat parametry mikroskopu, způsobem skenování i vzorkem samotným. Případně jeho přípravou na mikroskopování pomocí SEM. Při skenování lze například měnit urychlovací napětí. Urychlovací napětí ovlivňuje hloubku pronikání elektronů do vzorku a rozlišení obrazu. Nutno podotknout, že při vysokých urychlovacích napětích může docházet k poškození povrchu vzorku, zejména pokud je tenký nebo citlivý na elektronové paprsky. Vyšší napětí vede ke snížení kontrastu a celkově tvorbě tmavších oblastí vlivem většího pronikání elektronového paprsku do vzorku, nižší ke tvorbě světlejších. Dále se náboj může v oblastech s necelistvým pokrytím vodivou vrstvou nastřádat, čímž vznikají další světlé oblasti. Výsledný snímek mohou také ovlivnit magnetické oblasti (vychýlení paprsku), nebo oblasti s určitou orientací. Orientace vzorku, který je skenován hraje velkou roli. Čím je vzorek hrubší, tedy má větší rozdíl v orientaci a struktuře oblastí, tím bude i výsledný topografický kontrast znatelnější. Důležitým jevem, který s tímto souvisí je edge effect. Souvisí s úhlem dopadu paprsku. Při vyšším úhlu dopadu dochází k většímu průniku paprsku do povrchových

oblastí, což způsobuje, že úniková vzdálenost elektronů směrem k jedné straně paprsku se zmenšuje a počet sekundárních elektronů emitovaných ze vzorku se zvyšuje. Tento vyšší výskyt sekundárních elektronů pak přispívá k výraznějšímu kontrastu a vzniku světlých oblastí snímku. Důsledkem toho mají okraje, jimiž jsou tenké vyvýšené oblasti, strmé povrchy, výstupky a hrany ve vzorku tendenci vypadat jasněji ve srovnání s širokými plochými povrchy. V kontextu nanovláken tento jev ovlivní například nerovnosti na nanovláknech, nečistoty na vláknech nebo zvýrazní nerovnoměrný tvar vláken. Poslední jev se týká tenkých vláken, která jsou více vysvícená, především kvůli jejich křivosti. Ta je u nich větší než u širších vláken a mají proto tendenci vydávat větší kontrast. [2]

Oba zmíněné typy kontrastů (kompoziční a topografický) jsou při analýze různých látek a materiálů podstatné. Je třeba ovšem jasně vymezit, že v rámci této práce se snažím o simulaci topografického kontrastu ve snímcích. Právě ten se využívá u analýzy průměru a celkového vzhledu nanovláken. Pro vizuální porovnání je na obrázku 2.2 uvedeno porovnání obou typů. Dovolím si také zmínit, že je možné oba druhy kombinovat v různém poměru (umožňuje-li to konkrétní model mikroskopu). Konkrétně dělat jejich rozdíl a součet.



Obrázek 2.2: Porovnání snímku s topografickým kontrastem (levá strana) a snímku s chemickým kontrastem (pravá strana). Poskytnuto Bioing TUL

2.4 Aktuální způsoby měření průměru nanovláken

Pro projekt, k němuž má práce přispět, jsou stěžejní především pracoviště TUL, která provádí měření tloušťky nanovláken. Právě zde se setkáváme s užíváním nástroje ImageJ. Jedná se o open-source software vyvinutý pro zpracování a analýzu obrazu. Je široce využíván v různých vědních oblastech, včetně biologie, medicíny a pro nás především – materiálových vědách. Nástroj ImageJ nabízí širokou škálu funkcí pro manipulaci s obrazy, včetně základních operací jako je zvětšování, ořezávání, úprava jasu a kontrastu, až po pokročilé techniky jako je

segmentace, sledování buněk a také měření objektů v obrazcích. ImageJ také nabízí pluginy, které poskytují další funkce a možnosti analýzy obrazu. [13, 14]

Pro konkrétní užití měření tloušťky nanovláken poskytuje ImageJ poměrně přátelské rozhraní a v podstatě navazuje na ruční měření tloušťky vláken. Uživatel nejprve vloží snímek ze SEM. Následně ImageJ umožní uživateli zavést si měřítko, ke kterému pak vztahuje jednotlivá proměření materiálu (v našem případě nanovláken) na snímku. V praxi je vždy zkoumána určitá oblast. Často je to okolí diagonály, kterou osoba provádějící měření do obrázku vloží. Pomocí myši si pracovník určí dva body úsečky (počáteční a koncový) a přístroj zaznamená tuto vzdálenost na základě pixelů. Dle měřítka určeného v úvodním kroku pak přepočítává relativní vzdálenost ve snímku na reálnou ve zvoleném měřítku se zvolenou jednotkou (typicky nanometrech). Výstupem jsou jednotlivé hodnoty průměrů nanovláken. Z těch se pak pomocí externích programů provede analýza. Tou je třeba tvorba histogramu a zjištění minimální, maximální, či průměrné tloušťky vláken. Přes snadné užívání a možnost získání užitečných informací ztrácí tento nástroj především na rychlosti. Měření je totiž pro pracovníky časově náročné a bylo by ideální ho optimalizovat. [14]

Za zmínku stojí také DiameterJ, což je plugin předešlého softwaru. Tento plugin vznikl se snahou urychlit měření vláken. DiameterJ byl testován na řadě digitálních syntetických obrázků, které napodobují segmentované obrazy vláken (bílé čáry na černém pozadí), a na SEM obrázcích ocelových drátů se známými průměry. Konkrétně na vytvořených 118 obrázcích s bílými čarami různých pixelových průměrů na černém pozadí (vytvořeno v Inkscape). Dále SEM snímky ocelových drátů. Použity byly dráty z nerezové oceli o třech různých průměrech s nominálně monodisperzním průměrem. Samotný algoritmus probíhá v několika krocích. Prvním je segmentace, kdy je rozděleno pozadí a popředí snímku. Obrázky jsou segmentovány několika způsoby (globálním, lokálním, metodami strojového učení, detekcí hran). Také je zahrnuto prahování, odstranění šumu a vyhlazení čar. Následně užívá ImageJ několik různých algoritmů a analýz (Super Pixel Diameter Determination, Fiber Diameter Histogram, Mesh Hole Analysis a Fiber Orientation). DiameterJ uvádí mezi své výhody rychlost a srovnatelnost výsledků s manuálním měřením. [15]

Na základě konzultací z praxe však pro účely analýzy nanovláken, se kterými je pracováno na KCH TUL není vhodný pro většinu vláken. Jeho použití je možné pouze na malou skupinu materiálů, a i tyto snímky je třeba speciálně připravit pro tento typ měření průměru.

Dalším z nástrojů pro měření průměru polymerních vláken je SIMPoly. Jedná se o nástroj vytvořený v jazyku Matlab. Jeho trénování a vývoj byl postaven hlavně na synteticky generovaných datech. Některé z těchto dat se pokoušeli o simulujíc vlákenných struktur, jiné byly pouze barevně odlišené čáry. Také byl následně použit na snímky vláken poly(lactic-co-glycolic acid). Což je stejný materiál, s kterým pracoval i nástroj DiameterJ. Je to právě nástroj DiameterJ a klasický ImageJ, s kterými byl SIMPoly srovnáván. Výsledky, kterých dosahuje, jsou srovnatelné s DiameterJ. Při samotném postupu je nejdříve zvoleno měřítko. Následně se provádí metody z oblasti zpracování obrazu. Jako jsou segmentace, skeletonizace vláken, práce s kontrastem, prahování, zbavení se šumu. Tímto postupem je snímek zbaven nejasných vláken, a naopak rozeznatelná vlákna jsou zvýrazněna. Ve vláknech se nalezne středová linie a na základě nenulových pixelů se určí poloměry vláken. Poloměry jsou pak vyneseny do histogramu, na který je proložena zvonová křivka. Na základě těchto hodnot je vypočtena

průměrná hodnota průměru a střední odchylka. Z výsledků na datech od tvůrců DiameterJ plyne, že metoda SIMPoly dokáže přesně změřit průměr pixelu s 10 % chybou mezi známým průměrem pro všechny syntetické obrazy. Nicméně zde opět dochází k obdobným omezením jako u DiameterJ. Těmi jsou velmi úzké zaměření na materiály. Na rozdíl od DiameterJ má také nástroj méně možností analýzy. [16]

Posledním z nástrojů pro automatizované měření průměru vláken je GIFT. Jedná se opět o nástroj rozšiřující na ImageJ. Uživatel může GIFT aplikovat na jednotlivý obrázek nebo celou sadu obrázků ve složce. Opět jsou využity postupy jako detekce hran a prahování. Poté je vytvořena sada vzniklá otočením, se kterou se pracuje. Filtr otevírací čáry izoluje horizontální segmenty hran na každém otočeném obrázku a měří vertikální vzdálenosti mezi těmito segmenty. Vertikální vzdálenosti se zobrazí jako histogram a pomocí Gaussova rozložení se určí průměrný průměr vlákna. Uživatelům jsou poskytnuty histogramy a surová data pro každý obrázek, a dále pak souhrnná tabulka se všemi průměrnými průměry vláken. Pracovníci, kteří pro projekt zpracovávali data ke kontrole, vždy změřili několik vláken a z nich stanovili průměr. Ten se pak srovnával s výsledky nástroje GIFT. Testování probíhalo na vláknech želatiny z roztoku různé koncentrace. Z měření pak vyplynulo, že snímky s jednotvárněji orientovanými vlákny a vlákny podobného průměru měří nástroj podobně dobře jako předešlé dva nástroje a manuálně měřená vlákna. Nicméně opět zde docházelo k problému při měření vláken s nepravidelnou velikostí, tvarem a orientací. V tom scénáři se nejvíce blížilo měření manuálnímu nástroj SIMPoly. [17, 18]

2.5 Abnormality a defekty ve snímcích SEM vláken

Jednou z hlavních výzev při vytváření dat, které imitují reálné snímky, jsou právě nestejnorodost a možné variace jednotlivých snímků. Kromě rozdílných materiálů a různého zvětšení, což vede k nerovnoměrnému rozložení vláken na snímku, hraje také roli vznik a variabilita abnormalit, či defektů. Ty mohou mít různé formy a příčiny, které jsou závislé na materiálu, výrobním procesu, přípravě pro mikroskopii a průběhu mikroskopie samotné. V následující části se proto budeme podrobněji věnovat tématu abnormalit, které jsem pro přehlednost rozdělil do tří skupin.

První skupina zahrnuje vady vláken, chyby způsobené při přípravě a nečistoty. Ku příkladu oskenovaná vlákna na sobě mohou mít světlejší částice, které mohou být způsobeny jak defektem ze zvlákňovaní, tak i z roztoku či naprašování vodivou vrstvou. Nestejná vodivá vrstva může být nanesena i na vláknech, která se překrývají. Některé oblastí pak svítí více než jiné. Je to způsobeno špatným odváděním náboje – místo je pak světlejší. Světlá místa také vznikají na koncích vláken a místech, kde je vlákno přerušeno. Profil nanovlákna může být také spíš páskový, a nikoliv kruhového průřezu. Při výrazné změně průměru vlákna se opět může tvořit místo s abnormálním kontrastem. Často se v rámci vlákna může vyskytnout kapka. Původ takových kapek může být zapříčiněn nečistotou, ale i defekty na vláknech samotných. Naprašování vodivou vrstvou může být v nižších vrstvách nedostačující a mikroskop pak nedosahuje dostatečné rozlišovací schopnosti.

Druhým z typů defektů a abnormalit je vliv nanovláken na sebe sama. Může docházet k nalepení jednoho vlákna na druhé, a tím pak dochází třeba ke zploštění profilu vlákna. Vlákna se mohou kromě podélného slepení také křížit. V takových případech se vlákna buď fyzicky

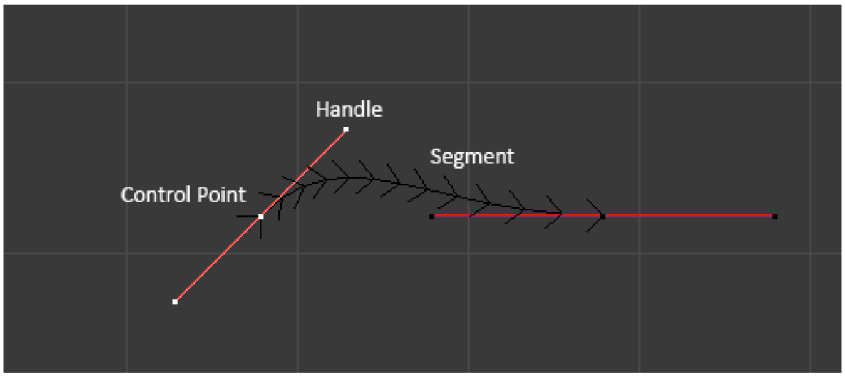
slijí, nebo je v oblasti křížení nedostačující rozlišení mikroskopu a vlákna jsou spojená jen zdánlivě. Spojování může nastat při dopadu na kolektor i během výroby. Často toto nastane není-li větší vlákno před dopadem menšího kompletně vysušeno. Pokud pak vlákna pokračují nějakou část své délky společně, působí jako jedno. V určitém bodě ovšem dojde k rozdělení. Tato vlákna se pro nejednoznačnost do měření v praxi nezahrnují. Nedochází totiž jen k změně kontrastu, ale i změně průměru.

Do poslední kategorie bych zařadil vliv skenování a polohy vláken. Zde je třeba zmínit hlavně vrstvení vláken. V některých případech může být horní vrstva od spodní tak vzdálena, že na spodní vlákna rozlišení mikroskopu nedostačuje. Na to jsou obzvláště citlivá vlákna menšího průměru. Samotná vrstva nanovláken není úplně rovná. Pro extrémnější případy pak vznikají vyvýšená světlá místa a nižší tmavá místa. V hlubších vrstvách snímku nemusí být vlákna dostatečně potažena vodivým prvkem, a tím vypadají tmavší než vlákna podobných průměrů v povrchových vrstvách. V opačném případě se může stát, že vlákno vystupuje nad povrch nanovlákenné vrstvy. Je tak více nabíjeno a více svítí. K podobnému jevu může docházet i na částech vlákna, a nikoliv na celých vláknech nebo oblastech snímku. Pokud v místě zesvětlení mění vlákno úhel (např. v útvarech připomínající koleno) a natáčí se směrem nahoru. Místo ohybu pak nemusí být tak dobře potaženo vodivou vrstvou, a tudíž se nabíjí, což způsobuje světlejší odstín jinak stejného vlákna. Při skenování může docházet k jevu, kdy je celý snímek tmavší. To může být způsobeno nedostatečným žhavením katody, kdy je proud elektronů menší. Může se to dít před koncem životnosti katody.

2.6 Výběr z 3D grafiky pro projekt a řešení v nástroji Blender

Aspekty vláken, snímků, vliv samotného SEM na ně a obecně celou vizuální stránku výstupu se v rámci projektu snažíme napodobit. K simulaci jsme původně přistupovali pomocí 2D modelu vláken, což je blíže popsáno v kapitole 3.1 Počátky a 2D přístup. Následně jsme z důvodů zmíněných také v této kapitole přešli na řešení formou 3D reprezentace. K té jsme použili program Blender. Pro projekt jsou důležité některé z pojmů 3D grafiky, se kterými právě při modelaci pomocí nástroje Blender přijdeme do styku. Věnovat se zde budu především světlům, oblasti objektů (včetně shader a mesh), na závěr pak render engine Cycles a EEVEE. Jelikož se práce týká právě nástroje Blender, je většina pojmů přejata a popsána na základě dokumentace [19].

V našem modelu jsou na základě hodnot ze skriptu generované modely nanovláken. Tyto vlákna reprezentujeme nejprve jako objekt typu curve (křivka). Takové objekty jsou geometricky modelovány pomocí tzv. spline křivek. Křivku lze definovat pomocí jednoho koncového bodu a vektoru do bodu následujícího. V rámci nástroje Blender jsou dva typy křivek Bézier a NURBS. Náš konkrétní model vlákna je tvořen pomocí typu Bézier. Hlavními prvky používanými při úpravě Bézierových křivek jsou řídící body a handles (rukojeti). Segment (skutečná křivka) je nalezen mezi dvěma řídícími body. Rukojeti určují zakřivení segmentu. Na obrázku 2.3 jsou řídící body nalezeny uprostřed růžové čáry, zatímco rukojeti tvoří prodloužení z řídícího bodu. Šipky vizualizují normály křivky, které například indikují směr a náklon. [20, 21]



Obrázek 2.3: Struktura Beziérovi křivky a její prvky v Blenderu. Převzato z dokumentace: https://docs.blender.org/manual/en/latest/modeling/curves/structure.html#bezier

Díky 3D modelovacímu prostoru prostředí Blender je možné měnit směr ve všech třech dimenzích. Každému z bodů křivky lze přiřadit poloměr, díky čemuž vzniká trojrozměrné vlákno určité šířky. Detail vlákna je závislý na množství bodů křivky, a také tzv. bevel resolution. Jedná se o rozlišení skosení pláště vlákna. Čím je vyšší, tím bude průřez vlákna více zakulacený a méně hranatý. Hodnota se odráží v počtu ploch (faces), které ho při převedení na síťový model (mesh) tvoří, respektive tvoří „prstenec“ mezi dvěma body křivky (viz dále).

V oblasti 3D grafiky a modelování se konkrétně setkáme s pojmem polygonální mesh. Jedná se o reprezentaci objektu, který modelujeme pomocí seskupených mnohoúhelníků. Uspořádání do takového celku je možné díky několika komponentům, které ho vytváří. V prví řadě jsou to vrcholy (verteces). Jejich primární účel je uchování souřadnic (typicky tři hodnoty). Kromě toho však mohou obsahovat i další informace jako barva nebo normálový vektor. Tyto vrcholy spojují hrany (edges). Vrcholy ani hrany nejsou v podstatě viditelné samy o sobě. Jedná se pouze pro vektorový způsob popisu umístění. Ve chvíli, kdy ovšem dojde ke spojení minimálně tří těchto objektů, dochází k tvorbě polygonu. Ten už je v rámci modelu možné vidět. Díky tomuto sledu všech součástí máme informaci o umístění, směrech a povrchu plochy. V 3D grafice je možné se setkat s různými tvary, nicméně dva základní jsou trojúhelník (nejminimalističtější) a následně plocha tvaru čtyřúhelníku. Tu lze ovšem taky reprezentovat trojúhelníkem (spojením dvojice trojúhelníků). Pomocí takto sestavených ploch pak lze tvořit povrch modelů a dále s ním pracovat (např. nanášet texturu). V případě struktury mesh vlákna je plášť tvořen čtyřúhelníkovými plochami. Je to vhodné kvůli podlouhlému modelu vláken. [20–22]

Modely vláken jsou potaženy materiálem. Konkrétní typy jsou popsány v kapitole 3.5. Materiály slouží především k volbě barvy modelu, případně nanesení textury, a nastavení chování při použití světel v modelu. V rámci nástroje Blender je pro úpravu materiálů určen celý pracovní prostor s názvem Shading. Zde je možné nastavovat základní věci týkající se materiálu, ale i specifická nastavení. Nastavení je možné spojováním uzlů v prostředí a ovládáním všemožných prvků materiálu. Konkrétní příklad, jak bylo s uzly pracováno jsou uvedeny v praktické části. Zde bych si dovolil zmínit základní věci ovládané na materiálu a spojení s osvětlením.

Před popisem samotného materiálu je třeba vyjasnit některé pojmy týkající se osvětlení. Obecně se v rámci 3D grafiky a modelování snažíme o realistické nasvícení simulující realitu. V projektu konkrétně je to simulace mikroskopování nanovláken v závislosti na různých

aspektech popsaných v přechozích částech. Složek světel je více a lze je popsat obecně. První je složka ambient, což je způsob osvícení celého prostředí. Jedná se o rozptýlené světlo, které nemá žádný konkrétní zdroj ani směr a rozprostírá se do všech směrů. Další složkou je diffuse. Pochází z jednoho směru od objektu rozptýleného do všech směrů. Tento jev závisí pouze na poloze světelného zdroje, nikoli na pozorovateli. Poslední z hlavních složek je specular. Odraz typu specular (zrcadlový odraz) pochází z jednoho zdroje světla, úhel dopadu je stejný jako úhel odrazu (plus rozptýlení). Závisí jak na poloze světelného zdroje, tak na pozici pozorovatele. [23–25]

Jako základ pro práci s materiálem slouží uzel shaderu Principled BSDF. Zkratka znamená Bidirectional Scattering Distribution Function (Funkce rozptylu světla v obou směrech) a jedná se o model nastavující shading materiálu. Jeho parametry ovlivní způsob, kterým poznamenají složky světla a barva finální vzhled materiálu. Je jím řízen vzhled a chování materiálu při procesu render. Už jen v tomto základním uzlu je možné nastavovat spoustu složek, proto zde vyzdvihnu především ty hlavní. Prvním parametrem je base color. Ten ovládá barvu, kterou lze zadávat v různých formátech (RGB, HSV atd.). V projektu je užit klasický model RGB s alfa kanálem (pro ovládání průsvitnosti). Nicméně vzhledem k šedotónové podstatě je využit zpravidla pouze jeden kanál. Dále je to trojice roughness (hrubost), metallic a index of reflection (IOR). Hrubost určuje drsnost povrchu, což souvisí se specular složkou a průhledností. Čím nižší je, tím se materiál jeví lesklejším, hladším a více odrážející světlo. Druhý parametr metallic funguje víceméně opačným způsobem. Maximální hodnota poskytuje kompletně zrcadlový odraz zabarvený základní barvou, a tedy maximum složky specular. Zároveň je při této maximální hodnotě potlačena difuzní složka a složka transmission. Při přesunu minimální hodnoty má materiál naopak tyto dvě složky a specular složka je pouze na vrchní vrstvě. Slouží tedy k optimálnímu rozvrstvení všech tří složek. Poslední parametr IOR je stupeň ohybu světelného paprsku, který prochází z jednoho prostředí do druhého. Je závislý na vlastnostech těchto dvou prostředí, konkrétně na poměrem mezi jejich indexy lomu. [20]

Pro získání snímků je třeba provést samotný render. Jako render se označuje proces transformace 3D scény do 2D obrazu. To zahrnuje výpočet osvětlení, stínování, odrazy, lom světla a dalších optických jevů. V projektu jsou užity dva typy render engine. Prvním je render, který slouží pro anotaci. Je to renderování z view portu a podrobněji je popsán v kapitole 3.6 věnující se anotaci. Jako hlavní je použit render engine EEVEE (extra easy environment engine) pro tvorbu snímků napodobující SEM. Jako jeho alternativu je v Blenderu možné použít také engine Cycles. Cycles je fotorealistický engine, který využívá techniku sledování paprsku (ray tracing). Ray tracing je technika v počítačové grafice, která simuluje chování světla ve scéně sledováním paprsků světla od kamery k objektům a zpět. Jeho výstup je obecně brán jako kvalitnější. Kvalita se ovšem projevuje i na potřebě dostatečně výkonného hardwaru. Příkladem, kde je možné zlepšení v kvalitě vidět, jsou stíny. Ovlivněna je také rychlost procesu render, a to hlavně u scén s velkým množstvím obsahu. Oproti tomu stojí EEVEE, který je užit i v této práci. Tento engine je postaven na OpenGL a jeho hlavní předností je rychlost. Obecně se EEVEE používá především pro rychlé náhledy. EEVEE není ray tracing renderovací engine, ale využívá proces nazývaný rasterizace. Rasterizace poskytuje interakci světla s předměty a materiály pomocí různých algoritmů. Cycles často poskytuje fyzikálně přesnější výsledky, a to právě kvůli rasterizaci EEVEE. Limitace EEVEE jsou v nižším množství nastavení materiálu

(a jeho chování), omezení v nastavení světel, kamery a stínech. I přes to jsem pro práci zvolil EEVEE. Toto rozhodnutí je podloženo v kapitole 3.8. [20, 26–28]

3 Praktická část

3.1 Počátky a 2D přístup

Pro vypracování projektu byl kolektivně zvolen jazyk Python. Nejprve jsme zkoušeli snímky analyzovat „ručně“ způsobem, který se snažil napodobit a navázat na postup užívaný v nástroji ImageJ na KCH. Skrz snímek byl proveden diagonální řez a hodnoty v tomto řezu byly analyzovány. S využitím analýzy hodnot jsme se snažili určit středy vláken a okraje vláken. Středy vláken mají většinou (ne však vždy) tmavší barvu, a tedy spíše nižší hodnotu takových pixelů. Okraje vláken bývají zpravidla světlejší a takové pixely pak nabývaly hodnoty vyšší. Porovnávání probíhalo v černobílých tónech na hodnotách 0 až 255. Myšlenka byla, že hodnoty řezu vláknem budou tvořit jakousi parabolu, ze které by bylo možné odvodit šířku vláken. Přes zkoumání grafů, jejich zpracování (porovnávání minim a maxim), odstraňování šumu atd. se však nepodařilo získat uspokojivé výsledky, a proto bylo od této metody upuštěno.

Navazujícím přístupem se tedy stalo tvořit si trénovací data samostatně. Data by následně společně s jejich anotací sloužila pro učení neuronové sítě. V základní verzi jsme vyšli s kolegou Davidem Šafaříkem ze skriptu pana doktora Pavla Mártona. Skript sloužil ke generování objektů imitující vlákna tzv. 2D cestou. Jednalo se o vykreslování pruhů v rovině. Tyto pruhy se skládaly z obdélníků, které měly okraje světlejší a směrem do středu barva tmavla. Tedy při proložení hodnot průřezu vznikala křivka podobná parabole. Díky tomu byl vytvořen 3D vzhled a pruhy tvořené obdélníky napodobovaly prostorový vzhled reálných vláken.

V základu byl skript doktora Mártona navržen k simulaci tvorby syntetických obrázků znázorňujících vláknité struktury. Využíval knihovny, jako jsou Matplotlib, NumPy, PIL (Python Imaging Library), SciPy, random, math a pickle. V začátku skriptu jsou definovány parametry a nastavení, především pro výpočet statistiky (calculate_statistics), počet obrázků k vygenerování (N_images) a parametry související se velikostí a charakteristikami syntetických vláken.

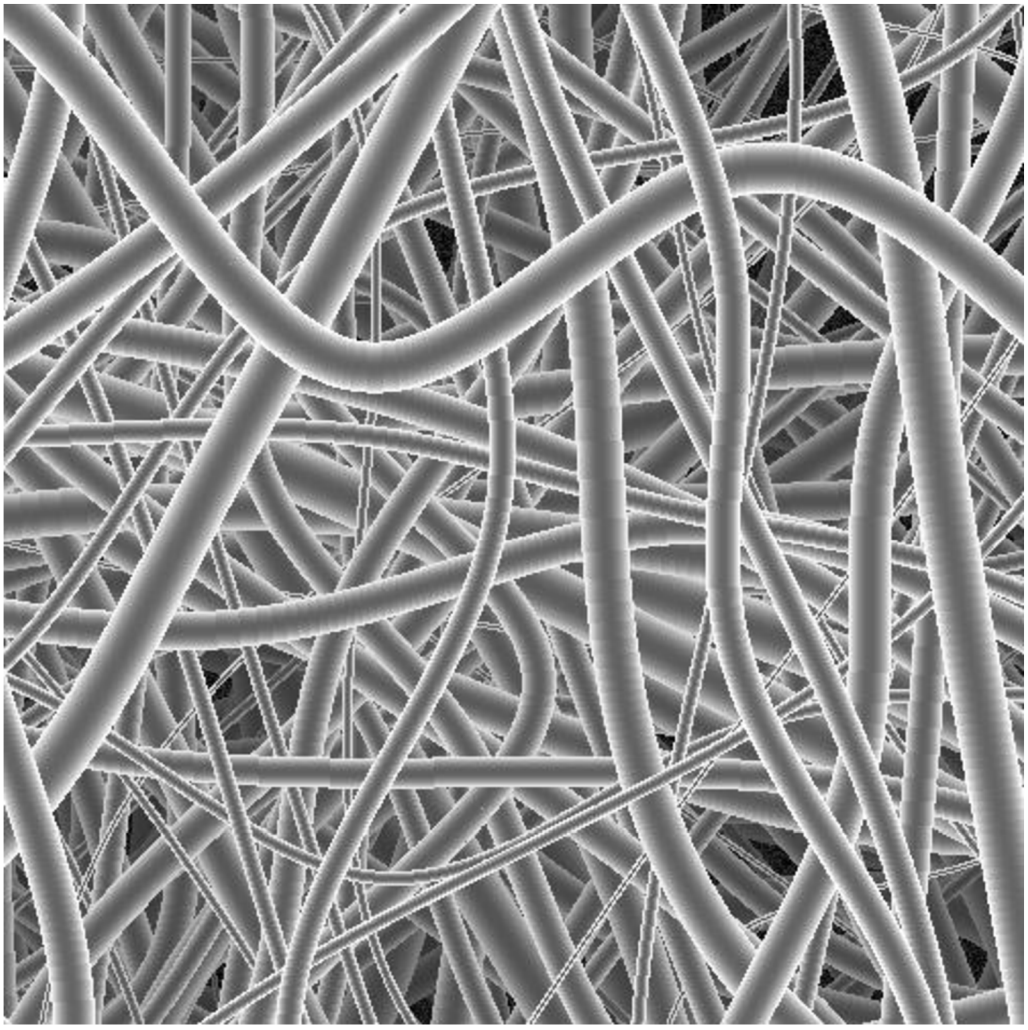
Následně proběhne iterace přes každý proces generování obrázku pro stanovený počet snímků. Pro každý obrázek se vytváří syntetická vlákna s náhodnými vlastnostmi. Těmi jsou především: délka, poloměr, počet míst ohybu vláken a úhel. S využitím těchto vlastností jsou vypočteny souřadnice bodů podél každého vlákna, přičemž je upravena hloubka a stínování v souladu s polohou vlákna. Tedy čím je vlákno hlouběji tím je tmavší. Výsledné obrázky jsou ukládány jako verze snímků SEM, tak i anotace segmentace.

Skript poskytuje možnost výpočtu statistik délek vláken s danými poloměry. Vypočítává délku a poloměr každého vlákna. Dále pak generuje histogramy, které zobrazují distribuci délek všech vláken a viditelných částí vláken ve vztahu k jejich poloměrům. Tyto statistické grafy jsou ukládány pro každý vygenerovaný obrázek. Snímky simulující SEM a segmentované snímky jsou ukládány jako pole NumPy pomocí pickle pro možnou další analýzu.

Pro další vývoj byl poměrně stěžejní matematický základ generování vláken této základní verze. Především po stránce ohybů vláken. Klíčová je ve skriptu interpolace, která se využívá pro vyhlazení křivek vláken. Po inicializaci několika bodů na křivce se používá interpolace,

konkrétně kubická interpolace, k vytvoření plynulého tvaru vlákna. Dále jsou pomocí goniometrických funkcí proveden výpočet souřadnic bodů, které tvoří vlákno. To zahrnuje výpočet posunu na ose x a y od středu vlákna podle daného úhlu a délky.

Co se samotné vizualizace vláken týče, je třeba zmínit jejich měnící se průměr. Každé samotné vlákno má jiný základní průměr, přičemž i tento průměr může být různý v různých bodech vlákna. A k tomu je opět užita interpolace. Tento proměnný průměr je poté použit k určení šířky vlákna v jednotlivých bodech. Pro vykreslení vláken s různou hloubkou a stínováním se používá matematický model, který kombinuje hloubku v referenci k ostatním vláknům, polohu a průměr vlákna. Hloubka zároveň simuluje nanášení vrstev nanovláken, a to, jak dopadají vlákna na sebe. Výstup základní verze skriptu je znázorněn na obrázku 3.1.



Obrázek 3.1: Původní snímky simulující SEM (autor Ing. Pavel Márton, Ph.D.)

Doktor Márton předložil solidní základní verzi, která sloužila jako robustní platforma pro generování dat. Nicméně, aby bylo možné dosáhnout věrohodných výsledků při napodobování skutečných snímků, bylo zřejmé, že je ještě před námi další práce. Je nezbytné dále upravit skript, aby bylo možné dosáhnout výsledků blížícím se realitě (skript je na GitHub, viz zdroj [29]).

Upravený kód byl v první řadě přepracován pro větší přehlednost. Došlo k několika významným změnám, které vylepšily celkovou funkcionalitu a čitelnost skriptu pro generování a vizualizaci SEM obrázků vláken. Importy jsou seskupeny na začátek souboru a uspořádány podle konvencí, což zvyšuje čitelnost a srozumitelnost kódu. Komentáře byly přidány k různým částem kódu, což poskytuje užitečné vysvětlení pro funkcionalitu a implementaci. Nejdůležitější změnou je modularita kódu, ve které byly vytvořeny třídy Fiber a SEMImage, a jejich metody ve skriptu. Tímto způsobem je kód lépe organizován a oddělen od ostatních částí projektu, což usnadňuje údržbu a správu kódu.

V třídě Fiber byla aktualizovaná metoda createFiber_PM_v0, aby lépe reflektovala požadavky projektu a poskytovala flexibilní možnosti generování vláken. Její parametry navazují na původní skript a slouží ke generování informací o vláknu. To je ukázáno v následujícím úryvku kódu.

```
def createFiber_PM_v0(self,
                      angle_in_plane=0,
                      dist_from_center=0,
                      bending_points=2,
                      bending_amplitude=0,
                      radius_min_max=(10, 10),
                      radius_variation_range=(0, 0.4)):
    """
    Creates a fiber with a random seed and a random bending points and a random
bending amplitude.
    The bending points and the bending amplitude are randomly generated.

    :param angle_in_plane: angle in plane of the fiber
    :param dist_from_center: distance from the center of the fiber
    :param bending_points: number of bending points
    :param bending_amplitude: bending amplitude
    :param radius_min_max: min and max radius of the fiber
    :param radius_variation_range: range of the radius variation in percent of the
fiber radius
    :return: None
    """
```

Zdrojový kód 3.1: Metoda createFiber_PM_v0 definující tvorbu vláken pro 2D model

Pro udržení kontroly nad náhodností jednotlivých údajů je použit seed. Při generování vlákna je aplikovaná obdobná matematická logika jako v základní verzi. Vlákno také mění svůj průměr po celé své délce. Upraveno bylo také vychylování centra vláken, čímž jsme dosáhli nepravidelné bílé barvy okraje u některých z vláken. To mělo zajistit rozmanitější typy vláken pro pozdější učení sítě. V rámci třídy vlákna je zajištěno generování parametrů vlákna a zachování klíčových informací o vlákně. Původní idea byla připravit více metod tohoto typu. Tím by pak bylo možné generovat snímky s určitými parametry založenými na reálných nanovláknech. Vznikla by tak možnost reflektovat různé materiály a postupy užívané v praxi.

Třída SEMImage byla také aktualizována, zejména metoda drawSingleFiber, aby lépe zohledňovala vlastnosti vláken a správně je vykreslovala. Byly přidány další výpočty a nastavení pro správné vykreslení a reprezentaci jednotlivých vláken v obrazu SEM. Třída SEMImage má nastavené parametry, které mají funkci zaměřenou na informaci o snímku samotném. Určuje se zde předně seed zmíněný v předchozím textu, počet vláken na obrázku,

možnost zavést do obrázku šum a odstín pozadí. Dále obsahuje několik metod, které slouží k tvorbě snímků vláken.

První klíčovou metodou je CreateFibers. V ní je vytvořena instance třídy Fiber a vytvořeno vlákno s údaji na jejím základu. Počet vláken je pochopitelně možno měnit dle požadavků na snímek. Následující dvě metody drawFibers a drawSingleFiber navazují na vytvořená vlákna a slouží k jejich vykreslení. Na vykreslení spolupracují obě metody, kde drawFibers řídí celkový proces vykreslování všech vláken a drawSingleFiber se stará o vykreslení jednotlivých vláken a jejich vlastností.

Metoda drawSingleFiber umožňuje vykreslení jednoho konkrétního vlákna. Nejprve se načtou informace o daném vlákně z pole fibers na základě zadaného indexu označující vlákno. Poté se vypočte faktor globální hloubky global_depth_factor na základě indexu vlákna vzhledem k celkovému počtu vláken a nastaví se základní barva vlákna base_fiber_color spolu s rozsahem exponenciálního profilu vlákna. Následně se iteruje přes jednotlivé segmenty vlákna. Pro každý segment se určí jeho poloměr a provede se iterace přes tři linie podél vlákna. Pro každý bod na dané čáře se vypočtou jeho souřadnice na základě polohy středu vlákna, směru paralelního a kolmého k vláknu a poloměru segmentu. Následně je vytvořen profil vlákna podle modelu paraboly. Poté se vypočte nová hodnota intenzity new_SEM_val na základě pozadí, globální hloubky a profilu vlákna a omezení se rozsah na hodnoty 0 až 1. Nakonec se aktualizují data snímku na příslušných pozicích a zaznamenávají se index vlákna, tloušťka a orientace na daných pozicích. V závěru metoda vykreslí okraje vlákna pomocí bodů na příslušných pozicích. Tímto způsobem tato metoda efektivně prochází jednotlivé segmenty vlákna a vykresluje je na SEM obrázku, zohledňující jejich polohu, směr a tloušťku.

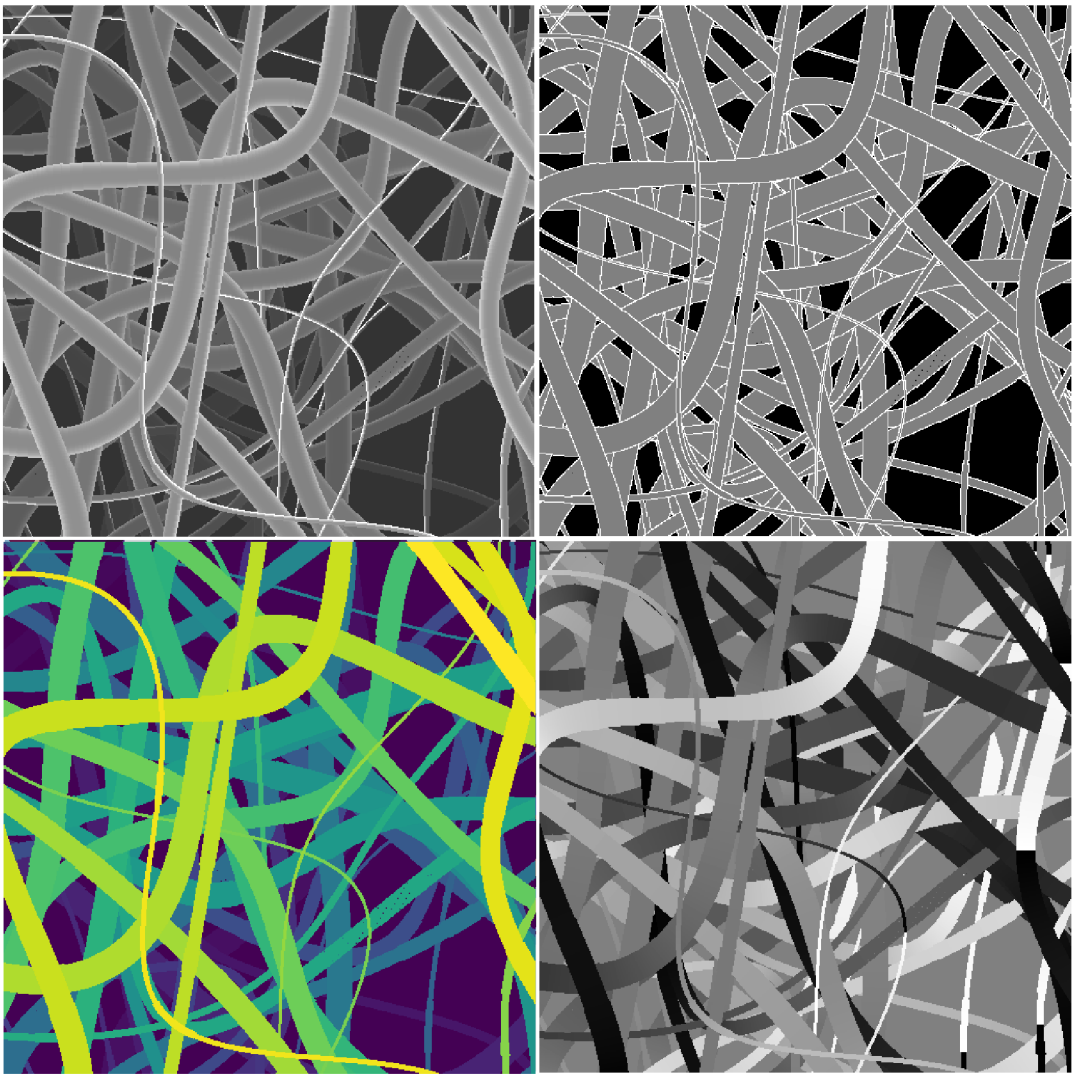
Metoda drawFibers slouží k vykreslení všech vláken na SEM obrázku spolu s přidáním šumu a nakreslením okrajového rámečku. Nejprve se vytvoří pozadí obrázku data_SEM jako pole s rozměry dvojnásobku okrajů plus rozměrů samotného obrázku. Dále se vytvoří pole pro indexy segmentů (data_SEG), tloušťku vláken (data_THI), hrany vláken (data_EDG) a orientaci vláken (data_ORI), která se všechna nastaví na výchozí hodnoty. Poté následuje iterace přes všechna vlákna ve zvolené oblasti. Pro každé vlákno se volá metoda drawSingleFiber, která vykreslí jednotlivé segmenty vlákna. Po dokončení vykreslení všech vláken se generuje šum (data_NOI) pomocí náhodného čísla v daném rozsahu a hodnota tohoto šumu se přičte k data_SEM. Poté se hodnoty data_SEM omezí v rozsahu 0 až 1.

Na to vše navazuje metoda saveFigure. Metoda saveFigure slouží k uložení vykreslených obrázků, které byly vytvořeny pomocí metod pro vykreslování vláken. Tak je učiněno pro všechny typy snímků. Těmi jsou SEM, SEG, THI, EDG, ORI. Užita je k tomu konkrétně knihovna matplotlib. Pro obrázky SEM a THI je použita šedá stupnice, pro ORI je použita colormap twilight a pro EDG je použita šedá stupnice s omezením hodnot na 0 až 1. Pro každý obrázek se nastavuje rozlišení DPI na 50.

Součástí skriptu jsou i metody calculateStatistics a analyze_shade. Jejich vznik byl založen na původní snaze o zjištění průměru vláken jen na základě analýzy snímků. Metoda calculateStatistics slouží k výpočtu statistik vláken na základě dat uložených v objektech vláken. Nejprve se vytvoří histogramy pro všechna vlákna a viditelné části vláken podle jejich poloměrů a délek segmentů. Pro každé vlákno se prochází jeho středy, délky segmentů a poloměry. Pro každý bod segmentu se zjišťuje, zda je uvnitř obrázku, a pokud ano, přidá se

jeho délka do příslušného histogramu podle poloměru vlákna. Nakonec se vytvoří graf, který zobrazuje délku vláken s daným poloměrem a délku viditelných částí vláken s daným poloměrem. Graf je uložen do souboru ve formátu PNG.

Metoda analyze_shade slouží k analýze odstínů na obrázku SEM. Nejprve se vytvoří instance třídy SemAnalyser pro daný obrázek. Poté se přidají dvě přímky o daném středu a poloměru. Následně se vytvoří graf zjištěných odstínů na obrázku a tento graf je uložen do souboru ve formátu PNG. Na obrázku 3.2 je možné vidět veškeré reprezentace generovaných snímků.



Obrázek 3.2: Finální reprezentace výstupu 2D řešení. První snímek simuluje reálné snímky z mikroskopu (SEM), druhý snímek odráží anotaci barvy pozadí, vláken a okrajů vláken (EDG), třetí snímek vyjadřuje vrstvy vláken segmentovaně (SEG) a poslední orientaci (úhly stočení vlákna) změnou odstínů (ORI)

Celý běh je dle standartní struktury řízen ve funkci main, sloužící ke generování a analýze obrázků napodobující SEM a ukládání výsledků do souborů. Nejprve se vytvoří složka pro ukládání výsledků. Poté jsou inicializovány seznamy pro ukládání výsledků SEM analýzy. Je určen počet snímků k vytvoření a iterací se přes tento počet pro každý obrázek vytváří instance třídy SEMImage s danými parametry. Na každém obrázku se generují a kreslí vlákna pomocí metod createFibers a drawFibers.

Obrázky se ukládají do souborů ve formátu PNG a provádí se analýza odstínů pomocí metody analyze_shade. Výsledná data se ukládají do příslušných seznamů pro pozdější použití. Pokud je nastaven příznak glob_calculate_statistics, provádí se výpočet statistik vláken pomocí metody calculateStatistics. Nakonec se všechna data uloží do souborů ve formátu pickle.

V průběhu vývoje jsme pracovali s různými odstíny a šumem v obrázku. Velká část úsilí byla také věnována okrajům vláken. Ta byla pro některá vlákna zcela bílá, jinde byl bílý pouze jeden z okrajů, a nakonec na některých generovaných nanovláknech bílé okraje chyběly úplně. Dále jsme také pracovali na změně průměru v rámci délky jednoho vlákna. Snímky zaznamenaly v průběhu vývoje nemalý posun. I přes to nedokázal 2D přístup dostatečně věrně zachytit reálný vzhled vláken. Přechod z generování 2D přístupu snímků simulujících vlákna pod mikroskopem na metodu využívající 3D pomocí dvojice Blender a Python byl nezbytným krokem k vylepšení realismu a komplexity naší simulace. Zatímco 2D přístup nám poskytoval určitou představu o struktuře vláken, nedokázal plně zachytit jejich prostorové uspořádání a interakce v různých hloubkách.

S přechodem na 3D simulaci jsme nyní schopni přesněji modelovat třírozměrné vlastnosti vláken, jako je jejich tvar, textura a prostorová distribuce. To nám umožňuje zkoumat jejich chování a interakce s větší přesností a realismem, což je klíčové pro pochopení zachycení chemických a fyzikálních vlastností. Celkově tedy přechod na 3D simulaci přináší významné výhody v porovnání s tradičním 2D přístupem, a to jak z hlediska realismu, orientaci v modelu, detailnosti simulace a flexibility práce.

3.2 Přechod na 3D nástroj

Při přechodu na tvorbu ve 3D bylo důležité zvážit různé nástroje a knihovny, dostupné ve spojení s Pythonem. Dále pro nás bylo důležité vybrat nástroj, který bude kompatibilní i v budoucnu. Hlavním požadavkem pak bylo, jak bude vhodný pro modelování nanovláken a zároveň tvorbu snímků. Několik následujících odstavců je věnováno zvažovaným nástrojům.

Začněme s Visualization Toolkit (VTK), který nabízí rozsáhlou funkcionalitu pro vizualizaci dat ve 3D [30]. Jeho předností je široká podpora funkcí včetně parametrických křivek, což může být užitečné pro tvorbu 3D modelů podobných vláknům. VTK poskytuje uživatelům širokou škálu nástrojů pro práci s daty a jejich vizualizaci, přičemž se věnuje i detailům a možnostem interakce s modelem. Přesto se jedná primárně o vizualizační nástroj, spíš než modelovací.

MayaVi, založený na VTK, působící jako jednodušší volba pro vizualizaci dat i ve 3D [31]. Jeho intuitivní rozhraní a užitečné funkce usnadňují práci s daty ve 3D. MayaVi poskytuje uživatelům možnost interaktivního průzkumu dat a manipulace s vizualizacemi, což je užitečné pro rychlé porozumění složitých datových struktur a vzorů. Nicméně jeho zaměření je také vhodný spíše na vizualizaci než pro tvorbu složitých 3D modelů, což by bylo omezením při potřebě více pokročilých a detailních funkcí pro tvorbu.

Dalším možným řešením bylo OpenGL, konkrétně PyOpenGL [32]. OpenGL je grafická knihovna sloužící k nízko úrovňovému programování 2D a 3D grafiky. PyOpenGL je knihovna pro Python, která umožňuje přístup k funkcím OpenGL prostřednictvím Pythonu. Díky této kombinaci je možné vytvářet vlastní 3D scény a efekty pomocí Pythonu.

Poslední variantou byl Blender [19, 33]. Jedná se o komplexní open-source 3D grafický software, který umožňuje tvorbu rozmanitých 3D modelů, animací a vizualizací. Díky svému integrovanému prostředí nabízí uživatelům široké spektrum nástrojů, včetně modelování, texturování, animace, renderování a kompozice. Pro tvorbu skriptovaných operací a automatizaci procesů lze v nástroji Blender využít jeho nativní podporu pro Python.

Python v nástroji Blender poskytuje možnost ovládat prakticky všechny aspekty softwaru, od vytváření a manipulace objektů a scén, přes nastavení osvětlení a materiálů, až po vykreslování scén. Tím, že Blender nabízí bohaté API a dokumentaci pro Python, umožňuje tvorbu uživatelsky definovaných doplňků, které mohou výrazně zvýšit efektivitu práce a přizpůsobit software konkrétním potřebám uživatele.

S ohledem na svou širokou funkcionalitu, komunitní podporu a možnost tvorby vlastních skriptů jsme si vybrali Blender jako nástroj pro tvorbu 3D modelů a vizualizací s využitím Pythonu. Automatizace prostřednictvím Pythonu umožňuje efektivní práci s modelem. Učení se ovládat Blender a programování i přes časovou náročnost, zejména pro začátečníky, se ovšem zdálo jako nejlepší varianta. Nakonec je nutné dodat, že původně jsme v rámci projektu užívali Blender 3.6, následně jsme ale přešli na verzi Blender 4.0, na níž byl projekt zpracován. Pro navazující práci doporučuji držet kontrolu nad verzí softwaru Blender, ve které bude program spouštěn.

3.3 Základní struktura ve skriptu a základy modelu

Skript je výsledkem spolupráce s kolegou Davidem Šafaříkem a některé informace, především na úrovni modelu vlákna, jsou tedy jeho doménou. Budou proto v této práci zmíněny spíše stručně. I přes to je důležitý určitý nadhled pro čtenáře této práce a z tohoto důvodu bude věnováno několik řádků právě popisu rozložení kódu, běhu skriptu, formování objektů a nastavení dalších struktur klíčových pro výsledné snímky s anotací. Parametry nejsou často právě v sekcích, jejichž autorem je David Šafařík rozebrány dopodrobna. Pro větší hlubší porozumění se odkážu na jeho práci.

Skript je dělen dle standartních postupů do několika částí, kterými jsou: import užívaných knihoven, definování tříd (s obsahem parametrů a metod jim vlastním), definování užívaných funkcí a proměnných, a na závěr samotný kód pro řízení tvorby modelu a generování snímků.

V rámci importovaných knihoven je třeba přednostně zmínit knihovnu bpy. Bpy poskytuje rozhraní pro interakci prostředí Blender pomocí jazyka Python. Je klíčovým prvkem pro automatizaci procesů v nástroji Blender a umožňuje vytvářet vlastní skripty. Pro výpočty dále slouží standartně používané knihovny math, NumPy a SciPy (z ní konkrétně interpolate). Kvůli různorodosti tvořených dat je třeba vložit do některých parametrů náhodnost. K tomu slouží knihovna random. Knihovnou pro kopírování v rámci kódu je užita knihovna copy.

3.3.1 Třídy a jejich metody

Program obsahuje tři třídy sloužící k nastavení a ovládání, jak budou snímky generovány. První třídou je CenterPointFiber. Jedná se o třídu uchovávající informaci o středových bodech tvořící páteř vlákna. Zahrnují údaj určující polohu v systému souřadnic x, y, z (stejnojmenné

parametry). Poloměr, které bude vlákno v daném bodě mít (radius). Směrové vektory kolmé (direction_perpendicular) a rovnoběžné (direction_parallel) k vláknu. Role třídy spočívá ve výpočtu pozice (ve 2D), délky segmentu (segment_length) a směrových vektorů pro další body vlákna, konkrétně metodou recalculate_attributes.

Navazující je třída Fiber. Ta uchovává informace o vláknech tvořených na základě středů ze třídy CenterPointFiber. Konkrétně obsahuje třída Fiber parametr seed používaný pro generátor náhodných čísel. Tento atribut umožňuje kontrolu nad náhodnými hodnotami generovanými v rámci třídy. Druhý parametr, i_dimension_L (ve směru x a y), určuje rozměr obrazu, pro který je vlákno generováno. Tento rozměr má vliv na celkovou velikost a umístění vlákna v rámci obrazu. Poslední parametr, segment_count, definuje počet středových bodů vlákna. Tato hodnota ovlivňuje detailnost a přesnost reprezentace vlákna v rámci obrazu.

Důležité jsou je obsah seznamu středových bodů vlákna (centers). Tyto středové body jsou inicializovány pomocí instance třídy CenterPointFiber (viz výše). Také využívá radius_base, sloužící k uchování základního poloměru vlákna (konstantní podél vlákna). Jsou důležitá při výpočtech týkajících se anotace a dalších částí programu.

V rámci třídy jsou dále definovány dvě metody. První z nich je metoda createFiber. Jedná se o metodu pro tvorbu vlákna s různými parametry, které ovlivňují jeho vlastnosti a vzhled. Mezi parametry patří horní mez v ose z a základní z-ová souřadnice, a minimální a maximální poloměr vlákna (radius_min_max). Jednotkou je metr užívaný nástrojem Blender, kdy pro představu odpovídá jeden metr 256 pixelům v obraze (při velikosti obrázku 1024x1024). Rozsah vychýlení poloměru i vychýlení z-ové souřadnice vlákna lze taky měnit příslušnými parametry. Dále jsou to parametry zajišťující ohyb vlákna. K těm patří úhel v rovině vlákna (angle_in_plane), počet bodů ohybu vlákna (bending_points) a amplituda ohnutí (bending_amplitude). A konečně vzdálenost od středu vlákna (dist_from_center).

Tato metoda používá výpočty na základě zadaných parametrů k vytvoření geometrického tvaru vlákna v prostoru, kdy vytváří středové body vlákna s proměnlivými vlastnostmi, jako je poloměr a z-ová souřadnice, a následně provádí výpočty pro výpočet dalších atributů vlákna.

Základní část je do jisté míry založena na původních myšlenkách z původního skriptu. Hlavní roli zde hraje funkce linspace z knihovny NumPy pro tvorbu rovnoměrného rozložení a interpolace z knihovny SciPy. Dále pak práce s úhly pomocí goniometrických funkcí. Díky tomu je možné vlákno realisticky ohýbat v několika bodech jeho délky, ovlivnit pozici na všech osách a měnit i poloměr vlákna po celé délce. Všechny tyto vlastnosti vláknu propůjčují vzhled pozorovatelný i u reálných nanovláken a zároveň umožňují mít kontrolu nad všemi údaji o vláknu.

Finální třídou je SEM_model. Tato třída uchovává informace o celém modelu a vláknech v něm. Třída má hned několik parametrů, které jsou klíčové pro inicializaci a definici modelu SEM. Prvním z nich je opět seed, zajišťující reprodukovatelnost výsledků. Dalším atributem je n_fibers, který určuje počet vláken vygenerovaných v modelu. Tím se definuje hustota vláken v syntetickém snímku. Parametr dimension_L udává rozměr viditelné části modelu. Jedná se o rozměr ve směru osy x a y. Parametr max_z definuje maximální souřadnici podél osy z, na kterou může vlákno dosáhnout. Tím se vymezuje výška vláken v modelu. Nad tímto prvkem je důležité mít kontrolu i pro správné nastavení kamery. Parametr segment_count určuje počet

středových bodů použitých v modelu vlákna. Tím se ovlivňuje detailnost a přesnost reprezentace jednotlivých vláken. Větší počet kromě detailnosti také zvyšuje časovou náročnost pří generování. Posledním parametrem je grid_size, který určuje počet mřížkových oddělení, do kterých je rozdělený viditelný prostor modelu. Tato mřížka pomáhá k organizaci a efektivnímu zpracování prostoru modelu.

Stěžejní je pro třídu SEM_model metoda generate_fibers. Jak název napovídá, tato metoda slouží k tvorbě jednotlivých vláken dle parametrů pro snímek. Pro svou funkci užívá následujících parametrů. První parametr radius_base_range určuje rozsah náhodně generovaného základního poloměru pro všechna vygenerovaná vlákna. Tento rozsah může ovlivnit celkovou velikost poloměru vláken. Druhý parametr radius_variation_range, stanovuje rozsah pro náhodně generovaný profil variace poloměru po celé délce vlákna v procentech. Díky ní vznikají nerovnoměrné tloušťky vláken, což přispívá k větší realističnosti modelu. Třetím parametr určuje rozsah změn v profilu souřadnice z podél vlákna. Tato variace umožňuje modelovat různé výšky vláken v prostoru, což přispívá k diverzifikaci vzhledu vygenerovaných vláken.

Parametr fiber_smoothing_filter_length stanovuje délku vyhlazovacího filtru, který se použije na souřadnice vláken, k vyhlazení hrubých hran nebo nerovností ve vygenerovaných vláknech, což vede k hladšímu a přesnějšímu vzhledu vláken. Jedná se o vstupní hodnotu předanou do další metody smooth_fiber_along_z. Tato metoda aplikuje vyhlazovací průměrovací filtr v souřadnici z zadaného vlákna. Tímto procesem se dosáhne vyhlazení z-ové souřadnice v rámci modelu vlákna.

Dalším parametrem je number_of_fibers_cut_in_middle, který určuje, pro jaký počet vláken je provedena metoda (cut_fiber_in_middle) třídy SEM_model. Tou je možné udělat v určitém bodě vlákna přerušení, ovládat přesné umístění tohoto bodu a také jak moc plynulá budou tato přerušení.

Parametr number_of_fibers_with_droplet určuje opět počet vláken. Pro ty je provedena další z deformací užitím metody generate_droplet_on_fiber. Jedná se o dilataci několik středů pomocí Hanningova okénka. Zde je možno nastavit velikost útvaru kapky a její umístění.

Posledním z parametrů je number_of_fibers_with_Y_division. Tento parametr opět určuje na kolik vláken je aplikována metoda, a sice metoda fiber_y_partition. Ta zajišťuje rozdělení vlákna do dvou odnoží. Vstupními parametry jsou objekt vlákna, dále úhel mezi částmi vlákna (angle_degree) a place_of_partition_in_percentage, který určuje, v jaké části se rozdělení stane. Je zadáván v procentech a převeden na konkrétní vzdálenost užitím celkového počtu. Úhel je převeden na radiány pomocí Numpy. Nejprve je vytvořena kopie vlákna pomocí copy.deepcopy a následně je vlákno přidáno mezi objekty vláken. Index odklonu (rozdělení vláken) se spočte jako součin počtu segmentů (self.segment_count) s place_of_partition_in_percentage, načež se umístí na segment odpovídající tomuto indexu. Nové středy, kterými vede druhá odnož vlákna, vznikají odečtením středu rotace od původního směru. A nakonec vrací metoda nové vlákno obsahující nově vytvořené středy. Celý proces je názorně vidět na Zdrojovém kódu 3.2.

Vzniklá odnož je brána jako samostatné vlákno, které se uloží jako samostatný objekt. To má výhody jak po modelovací stránce, tak i po stránce napodobování reálných dat. Díky

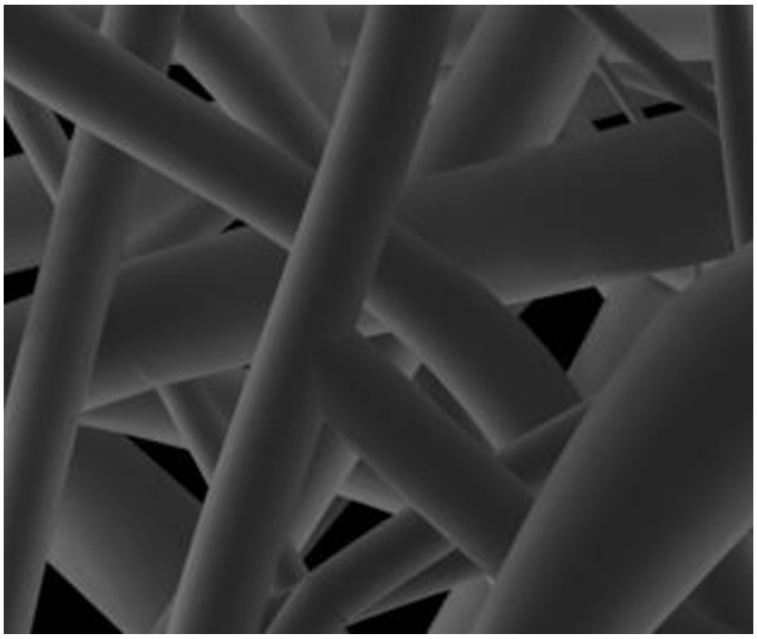
samostatnosti modelu vlákna je možné pracovat s jeho materiálem (materiály podrobněji v kapitole 3.4), ale také máme o vláknu veškeré informace, jako tomu je u ostatních vláken. V aktuální verzi má nová větev všechny údaje stejné jako „mateřské“ vlákno (poloměr, body ohybu, úhly v těchto bodech, …), až po vychýlení a následnou změnu souřadnic (x, y). V budoucím vývoji by díky znalosti všech údajů o vlákně bylo možné nové vlákno pozměnit od původního. Co se týká reálných snímků vláken, z tohoto hlediska dává také smysl, mít vlákno samostatně. Je tomu především proto, že ani v reálných snímcích nelze někdy zcela přesně určit, zda se vlákno rozdvojilo, nebo se třeba jen dvě vlákna slepila. Výsledek této metody je prezentován na obrázku 3.3.

```
 def fiber_y_partition(self, fiber, angle_degree,
place_of_partition_in_percentage=0.5):
    # Copy the fiber for partition
    y_fiber = copy.deepcopy(fiber)
    self.objects_fibers.append(y_fiber)
    # Convert degrees to radians
    angle = np.deg2rad(angle_degree)
    # Calculate the center of rotation for the whole fiber
    center_of_rotation = y_fiber.centers[int(self.segment_count *
place_of_partition_in_percentage)]
    # Construct a new "branch" of fiber
    new_centers = []
    for fiber_idx in range(int(self.segment_count *
place_of_partition_in_percentage), self.segment_count):
        x1 = y_fiber.centers[fiber_idx].x - center_of_rotation.x
        y1 = y_fiber.centers[fiber_idx].y - center_of_rotation.y
        x2 = x1 * np.cos(angle) - y1 * np.sin(angle)
        y2 = x1 * np.sin(angle) + y1 * np.cos(angle)
        y_fiber.centers[fiber_idx].x = x2 + center_of_rotation.x
        y_fiber.centers[fiber_idx].y = y2 + center_of_rotation.y
        new_centers.append(copy.deepcopy(y_fiber.centers[fiber_idx]))

    # Replace the centers list with a new one
    y_fiber.centers = new_centers
    y_fiber.segment_count = len(y_fiber.centers)

    return y_fiber
```

Zdrojový kód 3.2: Metoda number_of_fibers_with_Y_division pro tvorbu rozdvojení vláken

Generování vláken je inicializováno a následně provedeno na základě instance třídy Fiber a její metody createFiber. Právě té jsou předány potřebné parametry a je vytvořen požadovaný počet vláken pro snímek. Samotné generování je podrobněji popsáno v přechozí části.

V rámci třídy a tvoření vláken je také řešena kolize vláken (resolve_collisions), jejich překřížení (intersect) a napnutí vláken (stretch_fiber_between_collisions). A dále práce s umístěním v pracovní ploše (find_grid_position a assign_grid_positions). Tyto funkcionality obstarávají metody příslušné metody a přispívají k lepší simulaci reálných vláken. Kromě samotného generovaní vláken tedy metoda zahrnuje i další manipulace s nimi, díky kterým jsou vlákna tvořena kvalitněji a je možné zavést jim i různé defekty. To vše přispívá k realističnosti snímků.



Obrázek 3.3: Snímek zachycující rozdělení vlákna užitím metody number_of_fibers_with_Y_division (vlákna jednotné barvy bez šumu)

3.3.2 Užívané funkce

Na konečný chod programu má kromě výše uvedených tříd vliv i několik funkci stojících mimo ně. Většina těchto funkcí slouží ke správnému nastavení prostředí, přípravy scény pro renderování všech typů snímků, nebo pro manipulaci s objekty jinými, než jsou vlákna. Následující text bude podrobně věnován těmto funkcím a jejich úloze. Začněme funkcemi pro přípravu prostředí a umístění struktur do tohoto prostředí.

Před během skriptu je třeba vyčistit scénu a připravit ji pro generování následujícího snímku. K tomuto účelu je určena funkce clean_scene, která slouží k odstranění všech objektů, kolekcí, materiálů, textur, obrázků, křivek a všech dalších prvků ze scény. Nejprve se ověřuje, zda je aktivní nějaký objekt a zda není v režimu úprav (edit mode). Pokud ano, funkce přepíná tento objekt z režimu úprav zpět do běžného režimu, aby se zabránilo nečekanému chování. Následně se pro všechny objekty nastaví viditelnost, aby k nim bylo možné přistupovat. Je toho docíleno užití smyčkou procházející všechny objekty a nastavením obj.hide_set, obj.hide_select a obj.hide_viewport na hodnotu False. Všechny objekty jsou pak vybrány (bpy.ops.object.select_all(action="SELECT")) a vymazány (bpy.ops.object.delete) ze scény.

Pomocí smyček jsou procházeny kolekce (bpy.data.collections) a odstraňovány pomocí metody bpy.data.collections.remove na základě indexu smyčky. Obdobně jsou procházeny struktury world, pokud by došlo k modifikaci world shader. V závěru se vytváří nová struktura world pomocí bpy.ops.world.new() a přiřadí k se bpy.context.scene.world.

V závěru clean_scene je volána funkce purge_orphans. Tato funkce slouží k odstraňování tzv. sirotků neboli orphans objektů v aplikaci Blender. Jedná se o datové bloky, které zůstaly nevyužité v paměti, protože již nejsou spojeny s žádným objektem nebo scénou v projektu. Při spuštění se prochází všechny scény a datové bloky, a hledají se orphans objekty.

Pokud je verze Blender 3.0 nebo vyšší, použije funkci ops.outliner.orphans_purge z modulu bpy s několika specifickými parametry. Ty určují, že se mají odstranit lokální i propojené orphans objekty, a že se má provést rekurzivní prohledávání scén. Pokud je Blender verze nižší než 3.0, funkce se volá rekurzivně. Je ošetřena podmínkou a volá se, dokud nejsou všechny sirotci odstraněni. V práci je užita verze Blender 4.1, takže by zde měla být podmínka splněna.

Po vyčištění scény je možné umisťovat do pracovní plochy objekty. První z funkcí pro přípravu struktur do modelovacího prostředí je new_plane. Tato funkce je postavena nad funkcí z knihovny bpy pro generování primitivu typu plane (plochy). Prvním z parametrů je lokace. Jedná se o trojici čísel tvořící datovou strukturu tuple a sice souřadnice x, y, z. Dalším parametrem je size, tedy rozměr udávaný jako délka hrany plochy. Parametr name pak nastaví jméno objektu, pro případný přístup k objektu. Posledním z parametrů je plane_color. Jedná se o čtveřici čísel určující barvu plochy, a to v hodnotách RGB (první tři hodnoty) a alfa kanálu.

Průběh funkce je následující. Nejdříve je pomocí bpy.ops.mesh.primitive_plane_add vytvořen objekt plochy a to na základě parametrů location a size. Ten je následně vybrán z množiny objektů scény a je přiřazen proměnné plane (pro lepší manipulaci). Je mu přiřazeno jméno a nově vytvořený materiál. Materiál užívaný pro plochu je jednotný. Materiál nese jméno BlackMaterial, barvu dle parametru plane_material, hodnota roughness je 1 (určení hrubosti materiálu, viz dále) a specular_intensity je 0. Nakonec funkce vrací vytvořený objekt plane.

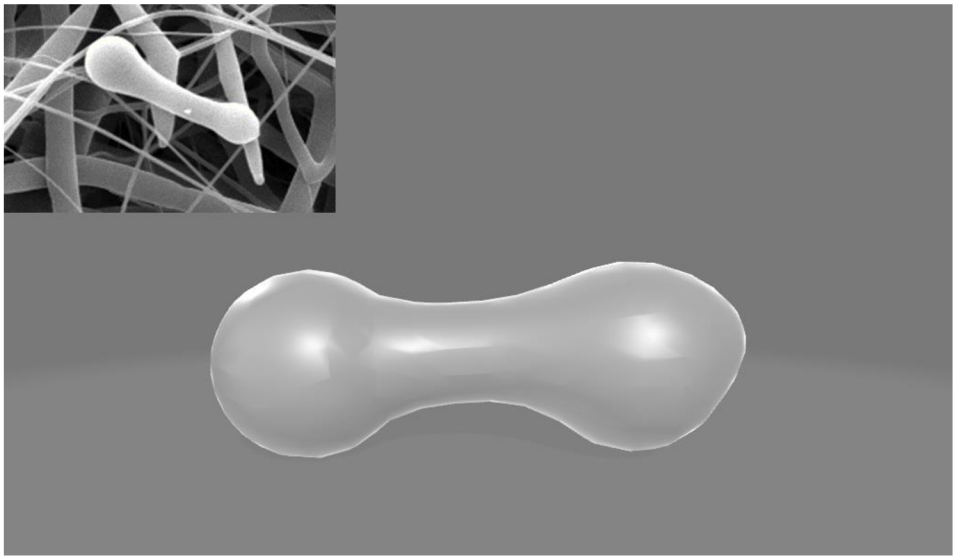
Tato plocha se používá jako podstava modelu, která ve finálním render snímků tvoří pozadí. Z toho důvodu je barva nastavena na hodnoty odpovídající černé v RGB (0, 0, 0) a velikost odpovídá záběru kamery s jedním metrem navíc (ortographic_scale + 1, podrobněji 3.4). Velikost je nastavena tímto způsobem, aby pokrylo pozadí ve finálních snímcích celou plochu (kvůli tomu rezerva ve velikosti). Její inicializace probíhá při přípravě SEM snímku funkcí sem_image_generating. Zahrnul jsem tvorbu plane v rámci této funkce, protože pozadí je nedílnou součást SEM snímku.

Dalšími důležitými prvky jsou nasvícení scény a kamera pro render snímků. K účelu nasvícení se používá hned několik typů světel. Jejich nastavení je provedeno pomocí funkcí create_light a apply_lightning_mode. Obě funkce spolu souvisí a navazují na funkci light_add přímo z modulu bpy. Vzhledem k důležitosti role světel jsou funkce a jejich vliv na scénu podrobně probrány v samostatné kapitole 3.7 Nasvícení SEM modelu. Nastavení kamery je pak zařízeno lokálně v řídícím kódu případně v konkrétních funkcích tvorby snímku SEM a anotací (viz dále).

Kromě plane tvořící pozadí, osvětlení a kamery je v rámci skriptu možné ještě vložit libovolný 3D model objektu s využitím funkce add_objects. Navazuje na funkci obj_import z knihovny bpy. Jako parametry je nutné uvést umístění objektu pro vložení (file_path), jméno pro objekt (name), souřadnice umístění (loacation_xyz) a změnu velikosti (scale_xyz). Souřadnice umístění a změna velikosti jsou vždy trojice čísel tvořící tuple. Souřadnice jako obvykle definují polohu dle souřadnic x, y, z. Změna velikosti v obdobných souřadnicích zvětšení či zmenšení objektu. Volitelně je navíc možné nastavit parametr random_placement na True. Tím je provedeno umístění i změna velikostí náhodně. Omezením je rozsah pro pozici 0 až 4 (souvisí s umístěním modelu vláken, kamery atd.).

Pro změnu velikosti je rozmezí při randomizaci 0,05 až 0,15. U změny velikosti je vždy třeba jednat dle vkládaného modelu. Co se jeho týká, je odzkoušeno import ve formátu OBJ (geometrická data 3D objekt) se souborem MTL (obsahuje materiálové informace).

Součástí projektu jsou i tři objekty vytvořené samostatně v nástroji Blender. Tyto objekty je možné vložit právě pomocí zmíněné funkce a slouží pro napodobení snímků s nějakou nečistotou, nebo jiný defekt snímku. Takové struktury nazýváme pracovně „non-fiber" objekty. Aktuálně projekt obsahuje projekt kuličku a podlouhlé útvary (viz obrázek 3.4). V dalším vývoji by bylo možné takových objektů vytvořit víc. Například krychličku napodobující prach, který je někdy přítomný v reálných snímcích.



Obrázek 3.4: Ukázka modelu defektu (abnormality) inspirovaná jedním z reálných snímků SEM. Předloha je umístěna v horním rohu obrázku

Asi nejdůležitější jsou funkce generující snímky. Snímky jsou obecně dvojího typu. První jsou SEM snímky, které právě simulují reálné snímky z mikroskopu. Druhým typem jsou snímky anotační. Ty právě slouží k předání informací o snímcích pro neuronovou síť. Anotační snímky a jejich tvorba budou podrobně popsány samostatně v kapitole 3.6 Anotace dat.

Generování SEM snímku obstarává funkce sem_image_generating. Ta přejímá parametry sem_model, use_single_color, single_color, metallic, roughness, IoR a use_noise_tex. Kromě sem_model, což je instance třídy SEM_model, se použijí všechny z uvedených parametrů k nastavení materiálu vláken. Manipulace s materiálem je jednou z hlavních oblastí zájmu pro docílení realističnosti vláken, a je proto podrobně popsána v kapitole 3.5 Materiál pro model SEM. Pro samotný chod funkce je obecně stěžejní nastavení view a barevnosti. Tento druh nastavení probíhá ve více částech kódu hlavně kvůli odlišné tvorbě anotace a SEM snímků. To probíhá v prvních řádcích kódu funkce, jak je vidět ve zdrojovém kódu 3.3. Jejich nastavení je pro optimální tvorbu snímku.

```
# Render scene color management - for SEM preferences
bpy.context.scene.display_settings.display_device = 'sRGB'
bpy.context.scene.view_settings.view_transform = 'AgX'
bpy.context.scene.view_settings.look = 'None'
bpy.context.scene.view_settings.exposure = 0
bpy.context.scene.view_settings.gamma = gamma_val
```

Zdrojový kód 3.3: Nastavení preferencí pro snímek ve funkci sem_image_generating

Nejprve je nastavena správa barev specifická pro zobrazovací zařízení na Standart Red Green Blue (sRGB). Následuje několik nastavení pohledu (view_settings). Prvním je nastavení view_transform na AgX. Jedná se o vylepšenou verzi Filmic (poskytuje lepší práci s barvami). Následně se nastavuje hodnota look na None. Tím je dosaženo standartního zobrazení výsledku bez filtrů a specifického pohledu. Expozice (exposure) je nastavena na výchozí hodnotu 0. Tím se v podstatě určujete, že na scénu nemá být aplikována žádná kompenzace expozice. To znamená, že jas scény bude zachován na výchozí úrovni bez jakýchkoli úprav pro přeexponování nebo podexponování. Hodnota gamma je v základu nastavena 1. Testovány byly i jiné hodnoty, kdy hlavní motivací byl vyšší jas v reálných snímcích. Vyšší hodnoty vedou ke světlejším snímků. Snímky ze SEM bývají také často světlejší, a proto se gamma stala jedním ze zkoumaných vlivů na výsledky z rozpoznávání snímků. Je možné ji nastavit pomocí proměnné gamma_val jako float od 1 do 5. Po zprocesování snímku SEM je nastavena zpět na hodnotu 1, aby nevznikaly nechtěné jevy v části anotační.

Po nastavení potřebných parametrů je připraven materiál pro vlákna. To se děje ve smyčce procházející objekty scény a kontrolující názvy objektů. Je-li součástí názvu objektu řetězec „fiber“, pak je vytvořen materiál. Materiály jsou v tuto chvíli jen vytvořeny. O tom více v další z kapitol. Co se samotných objektů vláken týče, ty jsou aktuálně typu CURVE. Což není vhodné pro další práci s nimi. Je tedy volána další smyčka procházející všechny objekty tohoto typu, která následně proběhne. Jako v mnoha dalších případech je to umožněno knihovnou bpy. Právě ta totiž udržuje informaci o všech objektech (bpy.data.objects). Ve chvíli kdy je splněna podmínka (object.type == "CURVE"), se nejprve najde v seznamu materiálů odpovídající materiál (vytvořený v přechozí části kódu). Poté je vytvořena reprezentace objektu jako mesh (geometrie objektu v podobě sítě, či mřížkový model), což je nezbytné pro další manipulaci s objekty, hlavně po stránce anotace. Práci se strukturou mesh je možné vidět ve zdrojovém kódu 3.4.

```
mesh = object.to_mesh()
# Create new object with the copied mesh
object_curve_mesh = bpy.data.objects.new(object.name + "_mesh", mesh.copy())
object_curve_mesh.matrix_world = object.matrix_world
bpy.context.collection.objects.link(object_curve_mesh)

# Set the curve object as the active object and delete
bpy.ops.object.select_all(action='DESELECT')
bpy.data.objects[object.name].select_set(True)
bpy.ops.object.delete()

# Select mesh object (fiber)
bpy.ops.object.select_all(action='DESELECT')
bpy.data.objects[object_curve_mesh.name].select_set(True)
```

Zdrojový kód 3.4: Práce se strukturou mesh vlákna ve funkci sem_image_generating

V prvních třech řádcích je vytvořen nový objekt, jako kopie vytvořené struktury mesh. Je pojmenován na základě původního vlákna s přídavkem řetězce „_mesh“ na konec názvu. Následně je zajištěno umístění a orientace nového objektu tak, aby odpovídal původní verzi vlákna. Nakonec je nový objekt zařazen do kolekce, a tedy i scény samotné. Původní instance je pak vybrána a vymazána. Nově vzniklý mesh je pak zvolen a je mu přiřazen materiál pro tvorbu snímku SEM. Obdobně pro všechny vlákna scény.

Tím je dokončen převod objektů na mesh a přiřazení materiálu. V závěru funkce je vytvořena nová plocha fungující jako pozadí pro snímek (viz funkce new_plane výše). Hloubka renderovaného snímku je nastavena na 8 bit, což odpovídá reálným snímkům z mikroskopu. Nastavíme velikost filtru pro render na 1,5 (filter_size), který používáme k vyhlazování obrazu a redukci aliasingu (rozostření a chyb v obrazu způsobených nedostatečným rozlišením). Čím vyšší je hodnota filter_size, tím větší je vyhlazování a tím méně aliasingu se objeví, bohužel na úkor detailů. Hodnota 1,5 je defaultní nastavení. Pro anotaci se ovšem přepíná na hodnotu 0, proto ji zde manuálně nastavíme. V souvislosti s anti-aliasingem je nakonec nastavena hodnota render_aa na FXAA. Jedná se o další způsob snížení nedostatečného rozlišení a vyhlazení hran. Konkrétně se jedná o Fast Approximate Anti-Aliasing. Tím je celý proces uzavřen.

Každá z anotací má vlastní funkci s podobnou strukturou jako uvedená funkce pro generování SEM snímku. Podrobně bude vysvětleno v další z kapitol. Nicméně co sdílí, je funkce pro render a uložení obrázků. Tu označujeme v kódu jako save_img. Jedná se o funkci se dvěma základními typy chodů. Parametry pro funkci jsou název (name), pod kterým má být snímek uložen. Druhým parametrem potom je index snímku (image_idx). Ten je předán jako číslo od 0 do požadovaného čísla snímků, což se specifikuje v řídícím skriptu. Posledním je příznak ano. V základu je nastaven False a určuje druh render. Při běhu funkce se nejprve nastaví output_path standartně ve složce out umístěné u skriptu. Zároveň proměnná určuje název snímku jako řetězec složený z indexu a jména uvedeného jako parametr. Nutné je také zmínit, že obrázek je uložen ve formátu PNG. Po nastavení cesty se pak podle příznaku renderuje snímek buď anotace (při ano=True), nebo jako snímek SEM (výchozí). Rozdíl je v režimu vykreslování scény. Pro anotaci je užit bpy.ops.render.opengl(write_still=True). Tento způsob vykreslí scénu OpenGL renderovacím engine. Odpovídá to náhledu viditelnému přímo v Blenderu. Díky tomu jsme schopni zachytit veškeré náležitosti anotací.

Render snímku SEM je pomocí bpy.ops.render.render(write_still=True, use_viewport=True). Zde se pro změnu jedná jiný render engine, a sice EEVEE. Tím je zachycen charakter vláken napodobující reálná vlákna. V obou případech je nastaveno write_still=True, což znamená, že výsledný obrázek obou režimů je zapsán na disk.

3.4 Řídící skript

3.4.1 Vstupní bod a prvotní parametry

Řídící část programu je umístěna v poslední části a v průběhu chodu využívá veškeré funkce a třídy zmíněné výše. Celý běh je znázorněn na obrázku 3.5. V první řadě proběhne nastavení proměnných. Těmi jsou texture_path, mesh_fiber_mode a number_of_images. První proměnná poskytující skriptu speciální texturu sloužící k anotaci. Tuto texturu je třeba vlastnit pro správný běh programu. Stejně jako složka pro výstupní snímky je tak nedílnou součástí programu. Další proměnná určuje, v jakém módu bude skript spuštěn. Aktuálně jsou v programu zahrnuty tři módy. První slouží jako standartní – obsahuje všechny podstatné funkce generování všech po sobě jdoucích anotací a také snímku SEM. Jedná se o plně funkční mód s momentálním stavem vývoje programu. Další dvě jsou experimentální – každá sloužila jednomu ze studentů k vývoji a experimentováním s modelem. Obecně jsme se snažili vývoj provádět v těchto dvou módech. Pro kompletní běh programu je tedy nastaven mesh_fiber_mode na 0. Tím proběhne standartní vykreslení všech snímků. Poslední dvojicí je proměnná number_of_images, sloužící k určení, kolik má program vytvořit snímků, a seed_int (číslo seed pro náhodnost) nastavený na 42.

Po nastavení těchto proměnných se spustí blok kódu obalený try. Pokud dojde k chybě je vypsána exception. V samotném kódu, je param_rand vytvořen jako instance třídy np.random.RandomState z knihovny NumPy s pevně nastaveným seedem na hodnotu seed_int. Tato instance param_rand je nyní připravena generovat pseudonáhodná čísla s pevně stanoveným seedem pro opakovatelnost výsledků. Program následně vypíše „Starting“ a následuje for loop, který proběhne pro každý z celkového množství snímků (pětice snímků SEM + anotace) definovaných proměnnou number_of_images v úvodu. Vypíše se název zpracovávaného snímku. Poté se pokusí vyčistit scénu voláním funkce clean_scene uvnitř bloku try. Pokud během vyčištění scény dojde k výjimce, program neukončí běh, ale místo toho se provede kód uvnitř bloku except. V tomto případě, pokud nastane jakákoli výjimka při čištění scény, program jednoduše vytiskne zprávu "ERROR: Could not clean scene." a pokračuje dál v běhu. Tímto způsobem se zajistí, že program nezastaví svůj běh kvůli chybě v procesu čištění scény, a umožní pokračovat v dalších částech kódu.

3.4.2 Nastavení parametrů scény a vláken

Následně se definují hodnoty proměnných v rámci snímku. Těmi je scale_ortho na hodnotu 4 určená k záběru kamery ve scéně (v metrech). Následně fiber_count definující počet vláken ve snímku (zvoleno výchozí na 50). Poté number_of_segments, což je počet segmentů vlákna. Tedy počet „prstenců“, ze kterých se utvoří finální model vlákna. Toto číslo ovlivní jak kvalitu modelů, tak i celkový čas vykreslení modelu. Pro vývojové účely jsme většinou pracovali

s 500 segmenty. Nicméně pro snímky, od kterých chce uživatel určitou kvalitu, bych doporučil 750 nebo i 1000. Momentální verze má tento počet nastaven na 750. Hodnotu je třeba volit také v závislosti na stroji, kde je program spuštěn. Vyšší hodnoty vedou k delší době tvorby modelu vláken, ale i lepším výsledkům. Dále maximální hodnota v z-ové ose nastavené na 3,5. Počet mřížkových oddělení grid_size na 12 (jako výchozí).

Výchozí rozsah šířky vláken radius_base_range je nastaven od 0,001 do 0,1. Především bych si dovolil zdůraznit spodní hranici, která je na poměrně nízké hodnotě. Je tomu tak proto, aby byly v obrázku dostatečně zahrnuta i tenká vlákna. Dále vychýlení od této hranice pomocí proměnné radius_variation_range. Procentuálně stanoveno od 0 do 100 %. Poslední je trojice proměnných definující množství vláken s určitým defektem (abnormalitou). Jsou jimi:

a) number_of_fibers_cut_in_middle – defekt přerušení vlákna uprostřed jeho délky
b) number_of_fibers_with_droplet – množství vláken s útvarem připomínající kapku
c) number_of_fibers_with_Y_division – defekt rozdvojení vlákna v místě jeho délky

Všechny tyto tři defekty jsou nastaveny na 1.

Na základě všech těchto proměnných je pak inicializována instance třídy SEM_model a volána její metoda generate_fibers. Té jsou taky poskytnuty předložené proměnné a dojde ke tvorbě vláken. Je vytvořena scéna pro umístnění modelu (scene = bpy.context.scene).

3.4.3 Náležitosti pro render snímku SEM

Postupuje se k první přípravě kamery. Výška kamery je nastavena staticky na 5 (camera_height). Její pozice je vypočtena jako polovina scale_ortho a uložena do proměnné camera_pos. Objekt kamery je pak přidán přímo pomocí knihovny bpy, která poskytuje funkci camera_add s parametrem location. Lokace je, jak bývá v Blenderu zvykem, určena dle souřadnic v soustavě x, y, z (location=(camera_pos, camera_pos, camera_height). Pro další práci ve skriptu je objekt kamery přiřazen proměnné camera.

Následně se provádí několik úprav nastavení scény a kamery v Blenderu pomocí knihovny bpy. Nejprve se komprese renderu nastaví na 0, aby nedocházelo ke ztrátě kvality a informace ve snímku (bpy.context.scene.render.image_settings.compression = 0), výstup se nastaví na černobílý obrázek (bpy.context.scene.render.image_settings.color_mode = 'BW') a hloubka snímku se nastaví na 16 bit (bpy.context.scene.render.image_settings.color_depth = '16'). Pokud by se tak neučinilo, snímky by obsahovaly nežádoucí šum. Kamera se nastaví jako aktivní pro vytvořenou scénu (scene.camera = camera) a její snímací mód na ortografický. Tím je zajištěn pohled bez perspektivy. To znamená, že objekty budou mít stejné měřítko a velikost bez ohledu na to, jak daleko jsou. K rozsahu kamery se následně využije proměnná scale_ortho. Díky tomu je viditelný čtverec o hraně 4 metry, zabírající celou část scény, kde je umístěn model vláken. Rozlišení renderu v pixelech je pak nastaveno pro snímky na 1024 v obou osách x a y. Pro jistotu je pak nastavena rotace kamery ve všech třech osách na 0. Výsledkem je kamera mířící směrem dolu v ortografickém módu, tvořící snímek s rozlišením 1024x1024 a v modelu zabírající model vláken.

Následně jsou nastavena světla funkcí apply_lightning_mode, a to je podrobně popsáno v sekci 3.7 Nasvícení SEM modelu. Po inicializaci světel proběhne nastavení zobrazení pixelových

souřadnic namísto relativních souřadnic v editoru UV (texturovacím editoru). Konkrétně se jedná o nastavení proměnné show_pixel_coords v třídě SpaceUVEditor knihovny bpy na True. Tím bude editor UV zobrazovat souřadnice v pixelech, což je užitečné při práci s texturami, protože umožňuje přesnější umístění textur.

Před práci prací s objekty vláken, která proběhne záhy, je ještě možné vložit do scény vlastní objekty funkcí add_object popsané dříve. Také je nastavena proměnná, ze které bude následně určen počet segmentů jednoho středu kostry vlákna (number_of_segments_of_bevel_circle). Toto číslo musí být sudé a odpovídá za finální zakulacenost vlákna. Tedy čím vyšší je hodnota, tím je prstenec tvořící vlákno zakulacenější (v momentální verzi výchozí hodnota 20).

Při tvorbě vláken se prochází smyčka skrze seznam objektů vláken. Kromě vlákna samotného je zaznamenán i index. To slouží k číslování vláken. Nejdříve je vytvořen název vlákna na základě indexu. Poté je pomocí knihovny bpy vytvořen nový objekt s právě vzniklým názvem typu křivky (curve_data = bpy.data.curves.new(name=object_name, type='CURVE')). Dimenze objektu jsou nastaveny na 3D. Pro optimální vyhlazení křivek je nastaven skrze vlákno tvořené jako instaci třídy curve rozlišení resolution_u na 1 a bevel_depth také na 1. Také je vypočten bevel_resolution tak, aby počet segmentů odpovídal hodnotě proměnné number_of_segments_of_bevel_circle. Díky tomu je zajištěna lepší kontrola nad modelem vláka. Vzniklá křivka reprezentující vlákno je pak přidána mezi objekty na základě těchto dat a jména.

Volitelně je pak implementována možnost zploštit vlákno pomocí funkce extrude. Tím je vlastně docíleno proužkového modelu, který odpovídá některým z reálných nanovláken. Jedná se o funkcionalitu, která není zcela odladěná v momentální verzi projektu a je spíše zájmem budoucího vývoje (podrobněji v práci D. Šafaříka).

Následující část kódu tvoří kontrolní body pro křivku a nastavuje různé vlastnosti těchto bodů. Nejprve se vytvoří nová spline křivka typu Bézier. Poté se do této křivky přidá určený počet Bézierových bodů, jejichž počet je určen délkou seznamu středů vlákna (fiber.centers). Následně se prochází každý prvek v tomto seznamu pomocí smyčky a pro každý bod se nastavuje jeho pozice na souřadnice získané z center.getCoords. Typ ovládacích bodů (handles) se nastavuje na VECTOR pro oba směry (levý i pravý). Poloměr bodu beziérovi křivky se nastavuje na poloměr získaný z center.radius.

Po tomto úseku jsou zakomponovány do skriptu tři možnosti match-case. Na základě proměnné mesh_fiber_mode se zvolí jeden ze tří výše uvedených módů. Pro účely práce bude nyní podrobně popsán standartní chod (mesh_fiber_mode = 0), protože zbylé dva jsou určené pro vývoj.

Po vstupu do bloku je volána metoda sem_image_generating, které je poskytnuta instance třídy SEM_model připravený dříve. Po doběhnutí funkce je celá scéna připravena na tvorbu prvního snímku SEM. Snímek je následně renderován a uložen pomocí funkce sem_image.

3.4.4 Příprava a samotný render anotačních obrázků

Po získání prvního typu snímku proběhne reset a nové nastavení náhledu. Toto nastavení je znázorněno ve zdrojovém kódu 3.5 a navazuje na nastavení popsané výše. Dovolím si ovšem ještě zdůraznit důležitost nastavení view_transform na Raw. Díky tomu lze pak pomocí Viewport render tvořit snímky anotace s korektním nastavením. Důležité je také nastavení hloubky barvy renderovaného snímku na 16 bit. To vede k zápisu plné informace.

```
# Render scene color management
bpy.context.scene.sequencer_colorspace_settings.name = 'Non-Color'
bpy.context.scene.view_settings.view_transform = 'Raw'
bpy.context.scene.display_settings.display_device = 'sRGB'
bpy.context.scene.view_settings.look = 'None'
bpy.context.scene.view_settings.exposure = 0
bpy.context.scene.view_settings.gamma = 1

# Change color depth - fixes noise in image should be 16 for annotation
bpy.context.scene.render.image_settings.color_depth = '16'
```

Zdrojový kód 3.5: Nastavení náhledu před tvorby anotace snímku

Po této přípravě začne série střídání jednotlivých anotačních funkcí a nastavení pro jejich korektní průběh. Samotné funkce anotace budou podrobně popsány v následující kapitole. Nastavení probíhá užitím knihovny bpy a pro podrobnější informace doporučuji hledat konkrétní funkce v její dokumentaci. První je anotace tloušťky nanovláken. Je tak učiněno za pomoci funkce thickeness_annotation, která po svém průběhu vrací thi_ori_mat. To je materiál sloužící pro anotaci tloušťky a orientace (viz dále).

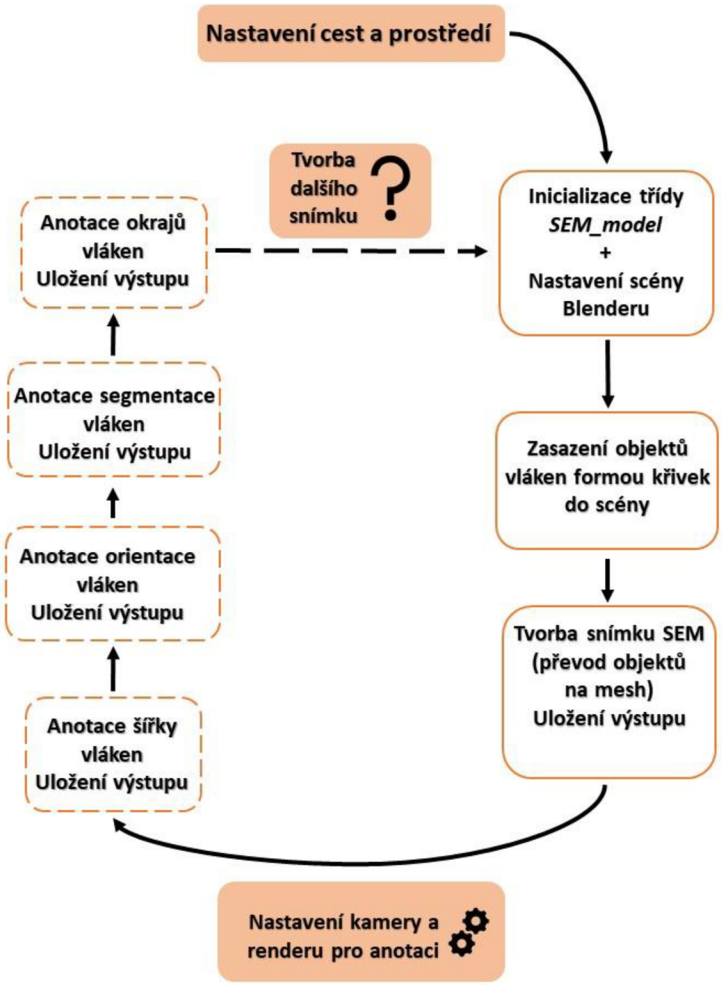
Následuje blok nastavující kameru. Nejdříve jsou pro render vypnuta světla ve scéně, aby neměli vliv anotační snímky (space_data.shading.use_scene_lights_render = False). Následně je nastaven průchod jen difuzním barvám do renderovaného obrázku. Tím se zbavíme odrazů a průhlednosti (space_data.shading.render_pass = 'DIFFUSE_COLOR'). Náhled kamery se pak nastaví tak, aby odpovídal výslednému renderu obrázku. Toho je docíleno přepnutím vieport shading mode na 'RENDERED' a získání momentálního náhledu. Také jeho následným přepnutím na kameru. Poté je vypnut anti-aliasing společně s nastavením velikosti filtru na 0. Tím se zajistí, že je správné renderování okrajů a dojde ke korektnímu zaznamenání průměru vlákna. Závěrem všech těchto kroků je volání funkce save_img s příznakem pro anotaci, aby proběhl render přímo z viewport.

Po tomto kroku je dokončeno obecné nastavení pro anotaci a pokračuje se k tvorbě dalších z anotačních snímků. Následuje volání funkce orientation_annotation, která připraví snímek pro anotaci orientace vláken. Výsledek jejího průběhu je uložen funkcí save_img s příznakem pro render anotace. Po orientaci je zaznamenána segmentace jednotlivých vláken pomocí funkce segmentation_annotation. Opět proběhne uložení pomocí save_img (s příznakem pro anotační render) a pokračuje se k poslední z anotací. Tou je anotace okrajů jednotlivých vláken. Tvorbu obstará funkce borders_annotation. Po jejím dokončení je uložen poslední anotační snímek pomocí save_img.

Tím je ukončen běh skriptu pro daný snímek. Na základě množství obrázků, které se mají vygenerovat dle proměnné number_of_images předané v úvodu, nastane jeden ze dvou

scénářů. Pokud už není nutné generovat další snímek, zavolá se funkce make_image_set_info program se ukončí. V případě, že je třeba vytvořit další snímky, vrací se program do úvodní sekce kódu. Vypíše se číslo nově generovaného snímku, dojde k vyčištění scény a celkově skript opakuje celý svůj běh. Tím je nastavovaní potřebných komponentů, tvorba snímků SEM a anotační část.

Volání funkce make_image_set_info je užíváno k záznamu informací o právě generované sadě. Jedná se o jednoduchou funkci, která jako parametry vezme všechny informace o sadě, které byly použity. Jsou jimi informace o vláknech, snímku, nasvícení, užitých deformacích, materiálu a případně využití šumu. Všechny informace jsou převedeny na řetězce a uloženy do textového souboru s názve img_set_info.txt. Jeho umístění je také ve složce out, aby byl pohromadě se samotnými obrázky. V souboru je také uložen čas a datum, kdy bylo generování provedeno. Tato funkce vznikla jako jedna z posledních součástí skriptu. Jedná se o praktickou součást programu a je především důležitá při zpětném zkoumání sad obrázků.



Obrázek 3.5: Schéma průběhu jednotlivých částí programu

3.4.5 Shrnutí datových formátů

Součástí zadání bylo také vybrat datový formát ve spolupráci s Davidem Šafaříkem. Následuje shrnutí datových formátů a práce s nimi v projektu. Informace o vláknech je uchována v rámci Pythonu v instancích tříd Fiber a SEM_model na základě zvyklostí objektového programování. Tyto třídy jsou reprezentovány v Blenderu pomocí 3D objektů vláken (fiber) typu CURVE a následně MESH. Výstup následně tvoří renderované obrázky dvojího typu. Snímky napodobující SEM jsou černobílé snímky o rozlišení 1024x1024 ve formátu PNG s hloubkou 8 bit. Snímky anotační jsou obdobné snímkům simulujícím SEM, ale v 16 bit hloubce a obsahem konkrétní anotované informace.

3.5 Materiál pro model SEM

Materiál v programu slouží dvěma hlavním účelům. V snímcích SEM je použit pro napodobení vzhledu reálných snímků. V rámci anotačních snímků pak přispívá ke korektnímu zachycení informace. Začněme u snímků SEM a jejich materiálu, protože v pořadí generovaných obrázků jen tento první.

Ve funkci sem_image_generating je pomocí smyčky procházen každý objekt a pokud má objekt v názvu „fiber“, proběhne tvorba materiálu. V dřívější verzi byl přiřazen jeden stejný materiál všem vláknům. V aktuální verzi je z názvu vlákna extrahován jeho index. Pomocí různých indexů jsou pojmenovány materiály a každé vlákno tak získá vlastní materiál. Nejdříve je sestaven název materiálu a ten je potom přiřazen novému materiálu. K tvorbě takového materiálu slouží knihovna bpy. Materiál je pak přiřazen objektu vlákna.

Blender při klasickém modelování užívá uzly (nodes), díky kterým je možné s materiály pracovat. Pomocí knihovny bpy je možné k uzlům přistupovat, propojovat je a nastavovat hodnoty jejich parametrů. Z tohoto důvodu je i v kódu vytvořen strom uzlů pro materiál a jako první je přidán uzel Principled BSDF, který v našem případě slouží primárně k nastavení barvy, metalicity, hrubosti a IoR. Celý průběh je možné vidět ve zdrojovém kódu 3.6. V rámci programu je možné použít v podstatě tři druhy materiálu.

Základním je materiál, který přiřadí každému vláknu stejnou barvu. V rámci parametrů je třeba nastavit příznak use_single_color na True a parametr single_color pak zadat jako čtveřici čísel, kdy první tři jsou klasické hodnoty RGB (rozsah od 0 do 1) a poslední je hodnota alpha pro průhlednost. Tento způsob je zde hlavně pro zachování možnosti jednolitých vláken z původní verze. Kromě barvy jsou při volání funkce zadány i parametry metallic (zvyšuje odlesk materiálu), roughness (řídí hrubost, respektive matnost materiálu) a IoR (řídí lom světla).

Dalším ze stylů je materiál, jehož barva je závislá na umístění vlákna. Vláknům, vytvořeným dříve, je přiřazena hodnota nižší (bližší černé), protože jsou ve spodní vrstvě modelu. Tím je simulována větší vzdálenost od kamery, která napodobuje snímající mikroskop. Naopak vlákna později přidaná, mají vyšší hodnotu barvy, a tedy i světlejší odstín. Konkrétní odstín je vypočten z celkového počtu vláken v modelu a vlastního indexu vlákna. Na závěr jsou pozměněny hodnoty metalicity, hrubosti a IoR. Zde je tvorba materiálu buď ukončena, nebo je ještě přidán šum. Materiál je funkcí přiřazován jednotlivým objektům až po jejich převodu na mesh.

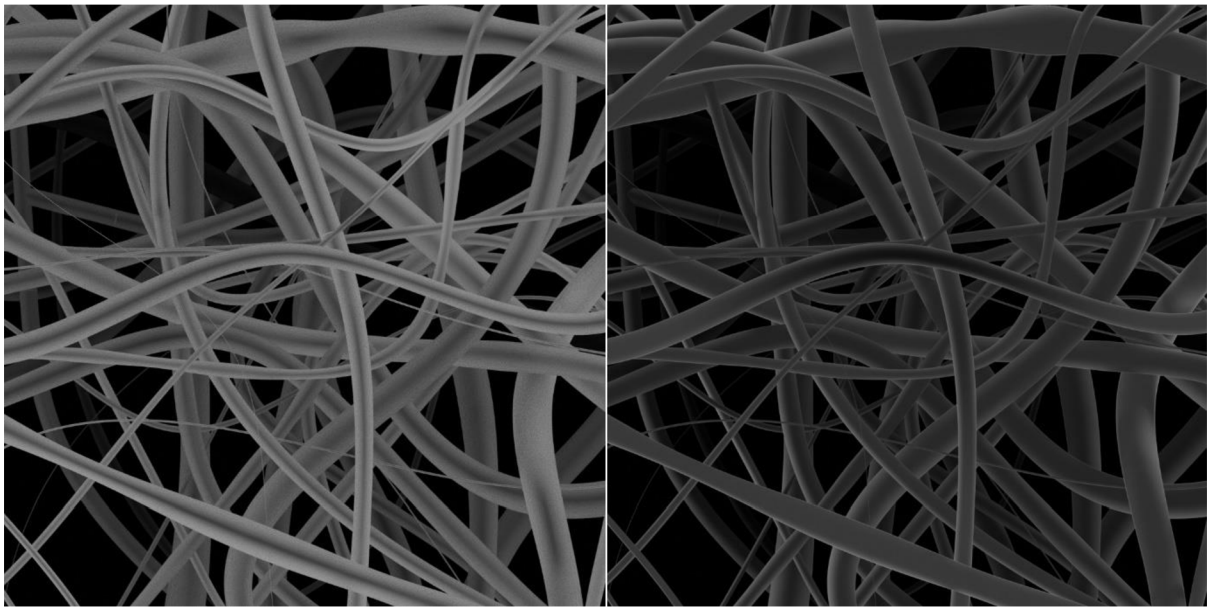
Pro přidání textury šumu je v parametrech funkce třeba přepnout příznak use_noise_tex na True. V takovém případě jsou pak nejdříve vytvořeny spoje pro nové uzly (links). Následně jsou vytvořeny uzly ShaderNodeTexCoord (texture_coords), ShaderNodeMapping (mapping), ShaderNodeTexNoise (noise_texture) a na závěr ShaderNodeBump (bump). Jejich propojení je v kódu 3.6 vidět na posledních řádcích.

```
principled_node =
bpy.data.materials[fiber_mat_str_name].node_tree.nodes["Principled BSDF"]
# Final color is calculated based on order of fibers
fiber_order_color = normalized_BW * (object_number_current + 1)
# Base Color (single or different per fiber)
if use_single_color:
    bpy.data.materials[fiber_mat_str_name].node_tree.nodes["Principled
BSDF"].inputs[0].default_value = single_color
else:
    bpy.data.materials[fiber_mat_str_name].node_tree.nodes["Principled
BSDF"].inputs[0].default_value = (fiber_order_color, fiber_order_color,
fiber_order_color, 1)
# Metallic
principled_node.inputs[1].default_value = metallic
# Roughness
principled_node.inputs[2].default_value = roughness
# IOR - Index of refraction for specular reflection and transmission.
principled_node.inputs[3].default_value = IoR
# Use noise on fibers material
if use_noise_tex:
    # Create links for nodes
    fiber_mat_links = fiber_material.node_tree.links
    # Create and set up nodes for shading
    texture_coords = nodes.new(type='ShaderNodeTexCoord')
    mapping = nodes.new(type='ShaderNodeMapping')
    noise_texture = nodes.new(type='ShaderNodeTexNoise')
    # Set scale of noise
    bpy.data.materials[fiber_mat_str_name].node_tree.nodes["Noise
Texture"].inputs[2].default_value = 100000
    bump = nodes.new(type='ShaderNodeBump')
    # Link nodes
    fiber_mat_links.new(texture_coords.outputs[0], mapping.inputs[0])
    fiber_mat_links.new(mapping.outputs[0], noise_texture.inputs[0])
    fiber_mat_links.new(noise_texture.outputs[0], bump.inputs[2])
    fiber_mat_links.new(bump.outputs[0], principled_node.inputs[5])
```

Zdrojový kód 3.6: Tvorba materiálu pro snímky SEM

Prvním uzlem je Texture Coordinate. Tento node zajišťuje souřadnice textur pro materiál. Určuje, jak jsou aplikovány textury na povrch objektu. Užit je první výstup Generated, který automaticky generuje souřadnice textury. Následně je napojen na uzel Mapping, kterému předá souřadnice jako vektor. Vektor je uzlem Mapping transformován a předán uzlu Noise Texture. Tento node generuje šumovou texturu, která se používá pro vytvoření nerovnosti na povrchu materiálu vlákna. Jeho výstupem je hodnota fraktálního šumu předávaná poslednímu uzlu Bump. Ten hodnotu přijme jako vstup pro výšku a vytváří na jejím základě nerovnosti na materiálu. Aby byly tyto nerovnosti viditelné, je zvýšena síla efektu mapování nerovností (strength) na 100 000. Tím vznikají interpolací velké změny mezi nejnižší a nejvyšší nerovností.

Pro porovnání je na obrázku 3.6 ukázán snímek s šumem a bez něj. Snímky obsahující šum jsou blízké reálným snímkům právě díky zašumění. To totiž do značné míry obsahují i reálné snímky. Jejich nevýhodou je ztráta pozitivního vlivu nasvícení obrázku. Skript samozřejmě umožňuje tvorbu obou typů snímků, což by mělo přispět k větší různorodosti trénovacích dat.



Obrázek 3.6: Porovnání snímku s texturou šumu (levá strana) a standartním materiálem (pravá strana)

3.6 Anotace dat

Pro projekt jsou všechny anotace realizovány jako obrázky vycházející z původního SEM obrázku. Anotační obrázky jsou v podstatě snímky stejných vláken nesoucí v každém z pixelů nějakou informaci. Momentální verze projektu anotuje průměr vlákna, orientaci, segmentaci jednotlivých vláken a vymezení okrajů vlákna (od těla vláken a pozadí).

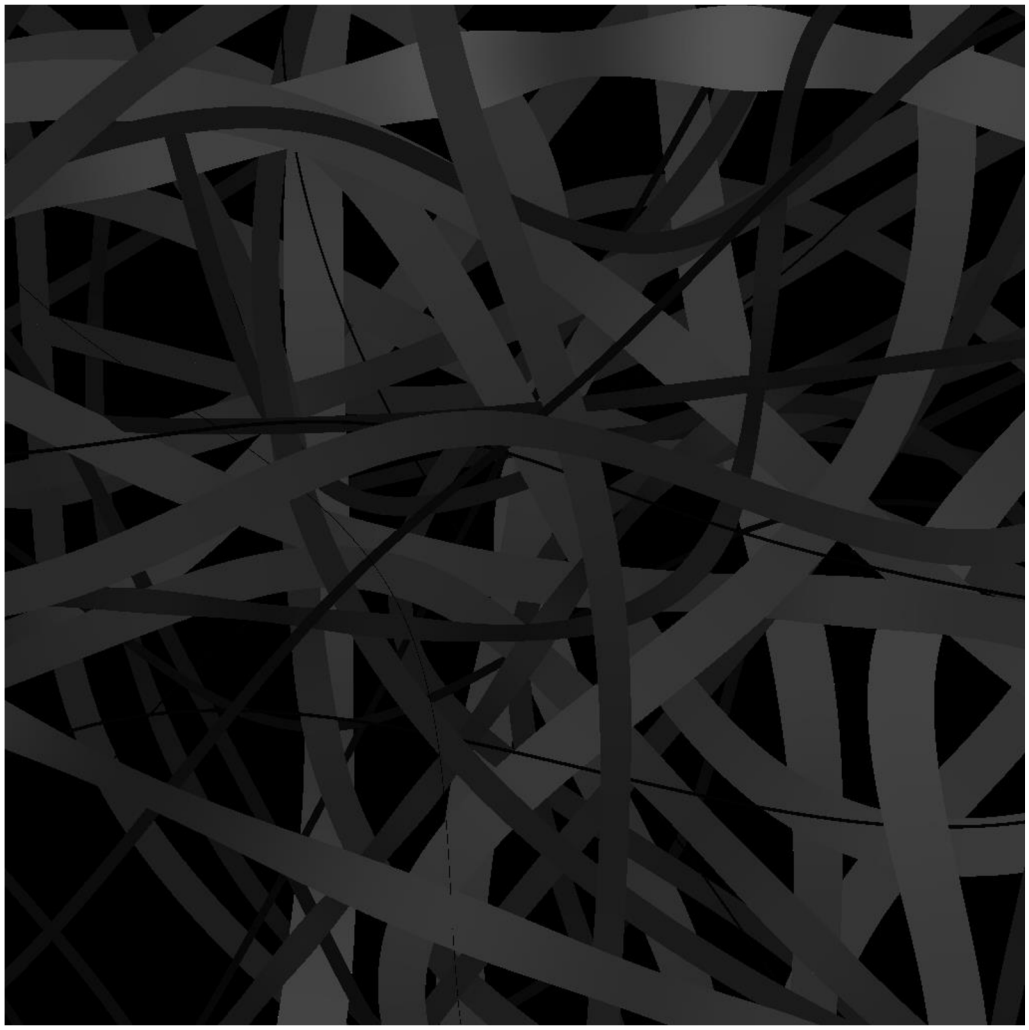
3.6.1 Anotace průměru (THI)

Tato anotace je realizována funkcí thickeness_annotation. Té jsou předány parametry sem_model (tj. instance třídy SEM_model), number_of_segments_of_bevel_circle, pixel_resolution a ano_texture. Jako pixel_resolution je předáváno rozlišení obrázku (tedy 1024). Pro parametr ano_texture je předán název textury speciálně navržené pro anotační účely. Jedná se o soubor „matlab_tex_256x256.png“, který je součástí projektu. Je to obrázek o velikosti 256x256 pixelů. Každý z těchto pixelů nese hodnoty mezi černou a bílou barvou. Díky formátu 16 bit je ve snímku 65 536 hodnot. Pixel v levém dolním rohu nese hodnotu 0 (černou). Hodnoty rostou po řádku až k pixelu 256. Poté další hodnota začne opět na novém řádku. Obrázek je využit jako textura a provádí se pomocí něj anotace (podrobněji níže).

Nejdříve je vytvořen nový materiál s názvem THI_ORI_mat. Následně jsou pomocí smyčky procházeny všechny objekty vláken podle indexu vlákna a původního materiálu. Ze SEM snímků je materiál nahrazen vytvořeným materiálem pro anotaci. Anotačnímu materiálu jsou

přidány uzly (nodes). Vytvoříme proměnné center_id a face_id a přiřadíme jim hodnotu 0. Pomocí center_id se zaznamená aktuální pozice centra (středu) vlákna. Pomocí face_id aktuální segment (plochy) tvořící kruh okolo centru vlákna. Postupně se prochází všechny plochy okolo středu, dokud se nedosáhne number_of_segments_of_bevel_circle. V tu chvíli se posune k dalšímu středu. V instanci třídy SEM_model je zaznamenána šířka okolo center, tedy poloměr (radius). Je tedy získán curr_radius, odpovídající aktuálnímu středu, a next_radius, odpovídající následujícímu. Z nich je vypočten průměrný radius. Tato hodnota se následně převedena na diskrétní hodnotu průměru vlákna (discreet_val) a to převodem na pixely. Výsledný integer je pak číslo odpovídající přímo hodnotě v textuře. Tedy jeden z 65 536 odstínů šedi. Odpovídající průměr se tedy do snímku propíše jako specifický odstín šedi mapovaný z přiložené textury jako UV mapy. Tenčí vlákna a jejich části jsou potom světlejší, zatímco širší vlákna jsou tmavší barvy. To je možno pozorovat na obrázku 3.7.

Po správném namapování je vytvořen uzel typu ShaderNodeTexImage. Ten je speciálně určen jako shader node pro předpřipravené textury. Pomocí něj je pak dle souřadnice UV mapy z přechozího kroku navázána textura na mesh vlákna. Pro korektní chod je ještě třeba nastavit cestu k textuře pro shader. U obrázku plnícího funkci textury nastavujeme načítací barevný prostor (colorspace_settings.name) na Non-Color. Dále se vypne snížení přesnosti dat nahraného obrázku (use_half_precision = False), tím budou hodnoty šedé v textuře korektně nahrány. Nakonec je nastavena okrajová mez textury na nulu (seam_margin = 0). Díky tomu nebude okraj textury skrze Blender nijak vyhlazován a bude ostře stanoven pro konkrétní obdélníky struktury mesh. Tím je textura korektně nahrána, přidána to shader uzlu a ten pak může být spojen s Principled BSDF uzlem pro materiál.



Obrázek 3.7: Anotace průměru vláken (THI). Zvýšená míra kontrastu, aby bylo možné vidět změny odstínů pouhým okem

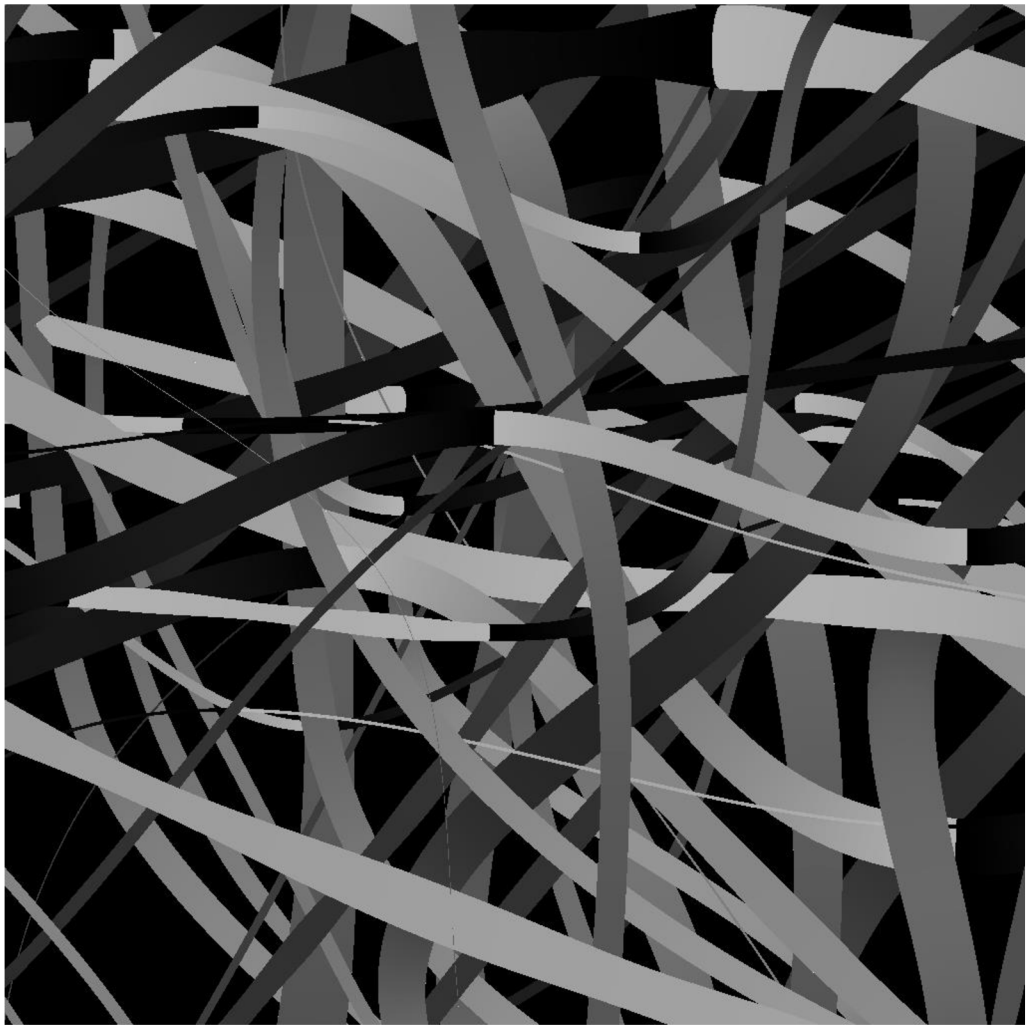
Pro tento uzel je nastavena hrubost na 0,5 a metalicita na 0. A pak především IOR na 1. Díky tomu nedochází ke stínování okrajů a může tak proběhnout anotace. Nalinkováním uzlů shader s texturou a principled je funkce thickeness_annotation dokončena. Funkce při ukončení vrací materiál pro anotaci tloušťky a orientace (thi_ori_mat). Následně je vše připraveno na render a pomocí funkce save_img je vytvořena anotace tloušťky.

3.6.2 Anotace orientace vláken (ORI)

Anotaci orientace se předává jako parametry instance třídy SEM_model a opět počet polygonů tvořící prstenec okolo středů vlákna (number_of_segments_of_bevel_circle). V podstatě je zde užitá stejná logika jako v předešlé anotaci s několika odlišnostmi. V úvodu je opět nastavena hloubka renderovaného obrazu na 16 bit. Následně probíhá smyčka přes všechny objekty. Pokud je objekt vlákno, je získán jeho index. Ten následně slouží jako indentifikátor v rámci instanci třídy SEM_model. Opět vytvoříme proměnné center_id a face_id. Nastaví se na hodnotu 0. Pomocí center_id se zaznamená aktuální pozice centra (středu) vlákna. Pomocí face_id aktuální polygonu tvořící prstenec okolo centru vlákna. Postupně se prochází všechny plochy okolo středu, dokud se nedosáhne number_of_segments_of_bevel_circle. V tu chvíli se

posune k dalšímu středu. Postupně se tak prochází středy konkrétního vlákna. Ty mají vždy přiřazen směrový vektor (direction_parallel). Se znalostí tohoto vektoru je vytvořena diskrétní hodnota, která bude hledána v obdobné textuře jako při anotaci tloušťky (discreet_val). Jedná se o úhel mezi vektorem a osou x. Výpočet proběhne jako arkus tangens podílu složky (curr_angle[1]) a složky x (curr_angle[0]) směrového vektoru. Číslo je převedeno na stupně a uloženo v datovém typu integer. Pro matematický výpočet slouží knihovna NumPy. Hodnoty následně upravujeme pomocí smyčky while na rozsah stupňů od 0º do 360º.

Hodnotu pak opět přiřadíme odpovídající souřadnici UV mapy anotační textury. Projdou se tedy všechny středy vlákna, a to postupně pro všechna vlákna modelu. Každé vlákno je zakončeno voláním object.update_tag. To zajistí aktualizaci viewport. Materiál zůstane stejný s pouhou změnou mapování textury na základě úhlů. Po skončení je výstup opět renderován voláním save_img. Tím je celá anotace zakončena. Výsledek je na obrázku 3.8.

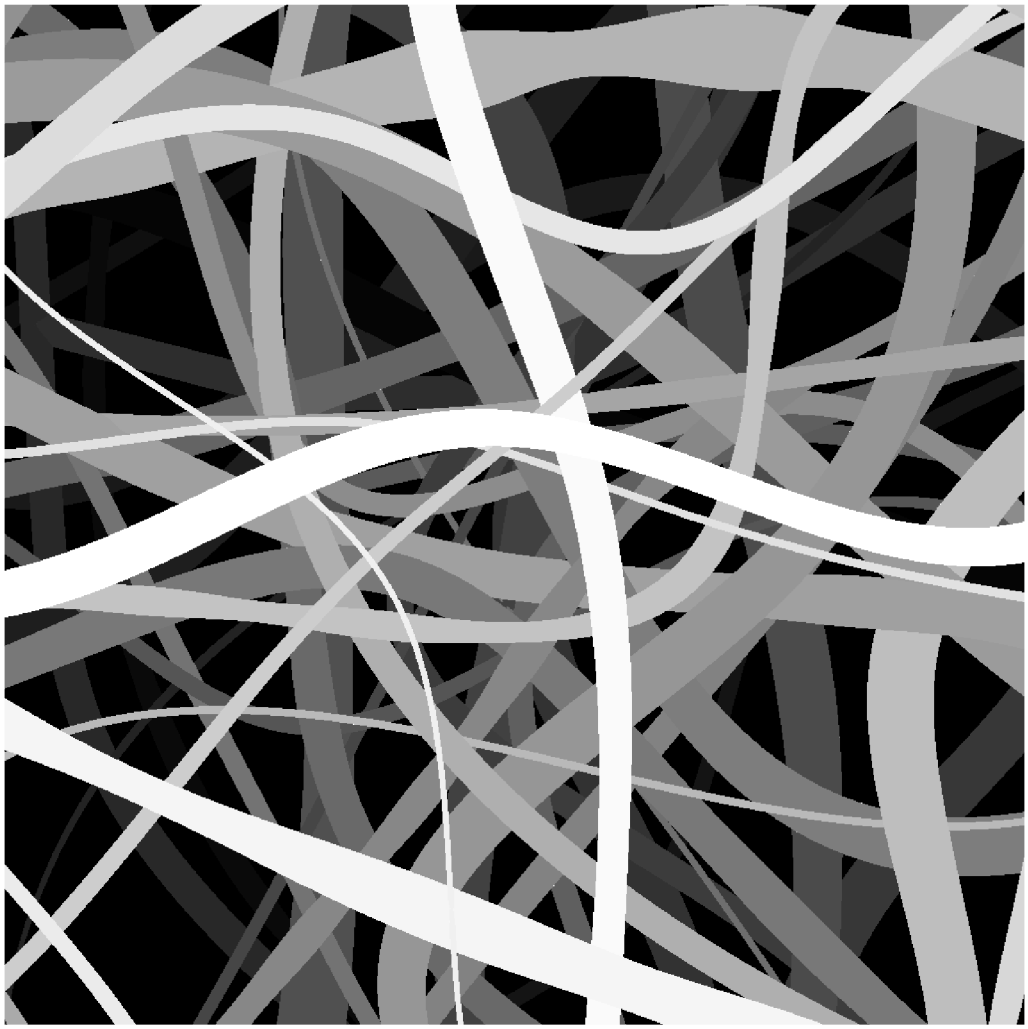


Obrázek 3.8: Anotace orientace (ORI) vláken. Zvýšená míra kontrastu, aby bylo možné vidět změny odstínů pouhým okem

3.6.3 Anotace jednotlivých vláken (SEG)

Tato anotace slouží k rozlišeních jednotlivých vláken (segmentaci) ve snímku. Je jí docíleno voláním funkce segmentation_annotation. Té jsou předány parametry material_to_replace a sem_model. Druhý z nich opět reprezentuje instance třídy SEM_model se všemi potřebnými informacemi o vláknech. Parametr material_to_replace je proměnná s materiálem, který je třeba vyměnit. V momentálním provedení je to thi_ori_mat užívaný v anotaci tloušťky a orientace. Samotný průběh je následující. Pro kontrolu je opět nastavena hloubka renderu na 16 bit. Následně je podle počtu vláken v modelu určeno číslo, které je uloženo jako proměnná fiber_count. Převrácená hodnota této proměnné je stanovena jako krok změny odstínu šedé při tvorbě barvy vláken (normalized_BW). Následně probíhá iterace skrze všechny objekty. Je-li objekt vlákno pak je určen jeho index z řetězce názvu vlákna (object_number_current).

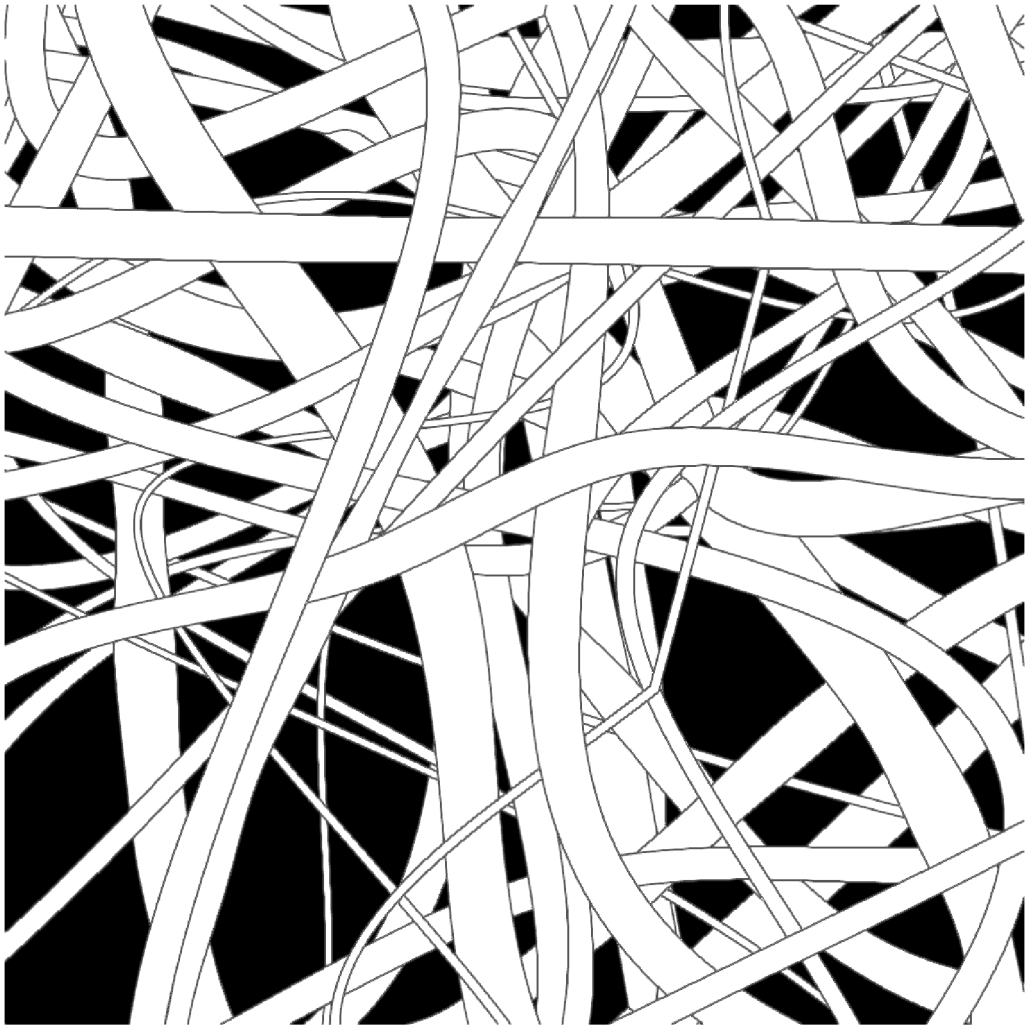
Podle indexu je pro každé vlákno vytvořen individuální materiál s názvem „SEG_mat_{object_number_current}". Tímto materiálem je pak pro každý objekt nahrazen původní materiál (pro THI a ORI). Pokud není u vlákna nalezen, je vypsáno upozornění, že nedošlo k úspěšnému nahrazení. V opačném případě se pro nový materiál segmentace vytvoří uzly. Opět se pro anotaci nastaví IOR na 1, aby nedošlo k rozmazání okrajů vláken a barva byla po celé oblasti vlákna stejná. Klíčovou je při této anotaci právě barva vlákna. Ta je obsažena v proměnné fiber_order_color. Je rovna součinu normalized_BW s indexem navýšeným o 1. Tím je jednak zajištěna změna barvy dle indexu zároveň pak odlišení od černého pozadí (vyhnutí se hodnotě 0 přičtením jedničky). Takto zaznamenaná informace je uložena jako ostatní snímky pomocí save_img. Výsledek je možné vidět na obrázku 3.9. Na závěr této anotace bych si dovolil zmínit, že při užití nedostatečného rozlišení snímků dochází k ztrátě informace vláken, a to konkrétně u vláken slabších. Momentálně se tomuto jevu v programu lze vyhnout pouze užitím vyššího rozlišení, nebo generování snímku bez extrémně tenkých vláken. Je pak spíše otázkou řešitele sítě, jak se s tímto problémem vypořádat. Doporučuji proto volit spodní hranici radius na 0,003.



Obrázek 3.9: Anotace segmentace (SEG) vláken. Slouží k oddělení jednotlivých vláken

3.6.4 Anotace okrajů vláken (BOR)

Poslední z anotací je označení okrajů vláken. A zároveň je v takto anotovaném snímku barevně vyhrazeno pozadí a těla vláken. Pozadí má standardně černou barvu, tedy všechny RGB hodnoty 0. Anotaci zajišťuje volání funkce borders_annotation. Jediným parametrem je fiber_color, což je čtveřice hodnot (RGB + alfa kanál) určující barvu vláken. Ve výchozím nastavení je tato hodnota (1, 1, 1, 1) odpovídající bílé barvě. Průběh funkce je následující. Nejprve je procházen seznam materiálů obsažených v modelu (bpy.data.materials). Pokud začíná název materiálu „SEG“, postupuje se dál. Tímto způsobem jsou vybrány veškeré materiály z přechozí segmentace. Té je pak přiřazena barva z parametru, a tím dojde ke změně barvy všech vláken na tuto barvu. Po projití všech materiálů se zruší označení všech objektů pomocí bpy.ops.object.select_all(action='DESELECT'). Na závěr se iteruje skrze všechny objekty, které mají v názvu „fiber“. Pro tyto objekty se nastaví příznak select_set takového objektu na True. Tím jsou označena všechna vlákna, na jejichž okraji z pohledu kamery vznikne označený okraj odlišný od barvy pozadí a těla vláken. Výstup je možné vidět na obrázku 3.10. Nakonec je třeba snímek pochopitelně uložit pomocí save_img.



Obrázek 3.10: Anotace okrajů vláken (BOR)

Všechny z anotací závisí na třech hlavních bodech. V první řadě, jednotlivé anotace je nutné generovat v určitém pořadí pro správnou funkci kódu. Za druhé je pro anotaci stěžejní, aby byl render a náhled správně nastaven. A nakonec aby byl využit viewport render (užívající jednodušší OpenGL). Součástí tohoto požadavku je pochopitelně korektní užití funkce save_img s příznakem pro anotaci. V posledním bodu bych zdůraznil práci s materiály. S těmi se pracuje skrze barvy a textury. Jako texturu je třeba použít obrázek, který je také přiložen v projektu, aby bylo zajištěno korektní namapování na síťový model vláken.

Mezi anotace by v budoucnu bylo možné zahrnout i další charakteristiky vláken. Nyní přichází v úvahu především anotace křivosti. Jedná se o charakteristiku závislou na průměru vlákna. Tím pádem by bylo možné tuto hodnotu relativně jednoduše vypočíst a zahrnout ji k ostatním anotacím. Jako způsob by bylo možné obdobně mapování na texturu jako v případě THI nebo ORI anotace.

3.7 Nasvícení SEM modelu

V reálných snímcích hraje důležitou roli jejich kontrast. Díky němu vzniká plasticita snímku a je tak zaznamenán topografický profil snímaného materiálu. Kontrast je základním požadavkem na snímek, protože bez něho nelze rozeznat jednotlivé útvary. A je to právě kontrast, který se v modelu snažím simulovat hlavně pomocí světel.

K nasvícení slouží primárně dvě funkce zahrnuté v programu. Prvně je to funkce create_light. Ta je rozšířením pro přidání objektu světla do prostředí Blender. Má hned několik parametrů, pomocí nichž jde specifikovat všechny potřebné údaje o objektu světla. V první řadě parametr type. Ten slouží k určení typu světla dostupných v nástroji Blender. Těmi jsou AREA (osvícení výřezu tvaru disk, čtverec, …), SPOT (sloužící pro osvícení kuželem světla podobající se baterce), SUN a POINT. Oba typy SUN a POINT nasvěcují celé své okolí, ale každý z nich jiným způsobem.

SUN emituje paprsky, které jsou rovnoběžné. Poskytuje světlo konstantní intenzity vyzařované jedním směrem z nekonečně velké vzdálenosti. Světlo z POINT se šíří rovnoměrně do všech směrů ze středu světelného zdroje. Intenzita světla klesá s kvadratickým zákonem v závislosti na vzdálenosti od zdroje. Rozdíly tedy jsou, že směr paprsků SUN je paralelní, zatímco POINT světlo vyzařuje světlo ze všech směrů, a liší se také ve tvorbě stínů. SUN světlo často tvoří ostré stíny, zatímco POINT světlo vytváří měkké stíny.

Kromě typu se dále specifikuje síla světelného zdroje. Výchozí hodnota je nastavena na 100. Pro světla POINT, SPOT a AREA je v jednotce Watt. Pro typ světla SUN je v W/m^2. Při tvorbě SUN doporučuji zadávat mnohem menší hodnotu, jak je tomu učiněno i v kódu. Dále je pro umístění do scény důležitá lokace, která je předána jako trojice čísel v tuple. V obou případech se jedná o řízení v souřadnicích x, y, z. Lokace (location_xyz) se zadává jako klasické souřadnice a rotace (degree_rotation_xyz) ve stupních. Dovolím si upozornit na možnost užívání i záporných hodnot. Dalším parametrem je pochopitelně barva, zadávaná jako tuple čtveřice čísel odpovídající RGB a alfa kanálu. Dále je předáván parametr radius, který určuje velikost sférické plochy, ze které je světlo vysíláno. Když je tento parametr nastaven na hodnotu větší než nula, světlo bude vysílat paprsky z této sférické plochy s daným poloměrem. Světla s větší velikostí mají měkčí stíny a lesklé odlesky, a také budou působit tlumeněji, protože jejich energie je rozložena na větší ploše. Pro výchozí běh programu je nastaveno na 4. Dalším parametrem je name pro určení jména světelného objektu ve scéně. A dál je zadán parametr align, který je nastaven ve výchozí formě na WORLD. Používá se k zarovnání více vybraných objektů tak, aby se zarovnaly na určené ose. Předposledním poměrně důležitým parametrem je shadow_eneabled. Ten slouží ke kontrole, zda světlo vrhá stíny. Ve většině případů projektu je tato možnost vypnuta. A poslední parametr se projeví jen tehdy, pokud je zvolen typ světla AREA. Jedná se o area_shape a řídí tvar tohoto světla. Možnosti jsou SQUARE (nastaven jako výchozí), RECTANGLE, SPHERE a ELIPSE.

Parametry radius, type, align a location_xyz jsou v začátku funkce předány metodě Blenderu light_add. Poté je tento objekt uložen do proměnné light, s kterou je pracováno v rámci skriptu. V následujícím kroku je provedena rotace dle zadaných hodnot úhlů v systému Eulerových úhlů (light. rotation_euler). Nastaví se energie světla (light.data.energy), barva (light.color) a název (light.name).

Pro všechny názvy objektů světel je sestaven řetězec začínající „L_“ a po něm samotný název. Tento způsob pojmenování jsem zvolil pro přehlednost řazení objektů v prostředí Blender. Předposledním krokem je vypnutí stínů světel (light.data.use_shadow). Ačkoliv tento krok může působit banálně, jedná se o důležitý krok. V našem modelu se snažíme simulovat mikroskopii svícením světlem. Nicméně reálné snímky vznikají pomocí detektorů, které registrují vyzářené elektrony, nikoliv nasvícení. Je to tedy spíše jako kdyby záření vycházelo ze vzorku, a nikoliv dopadalo na něj (jak je tomu v nasvícení scény). Následkem toho vznikají stíny při překrytí světelného paprsku. A přesně proto je třeba stíny vypnout. Ve finální části je pak jen provedena kontrola, zda je typ světla AREA. Pokud tomu tak je, pak získá světlo parametrem předepsaný tvar (light.data.shape). Když proběhnou všechny předchozí kroky, vrátí funkce vytvořené světlo.

Práce se světly zpracována ve funkci apply_lightning_mode. Funkce má čtyři parametry a plní řídící roli všech světel ve scéně. Prvním parametrem (mode) je řetězec názvu módu osvětlení, který se má ve scéně spustit. Dalším parametrem je random_spotlights, a pokud je nastaven na True, je do snímku vloženo světlo typu SPOT. S ním souvisí parametr spot_seed. Jedná se o seed pro řízenou náhodnost umístění světla typu SPOT (viz dále). Obdobně lze na True nastavit poslední z parametrů, kterým je add_center_sun. Ten umožňuje přidat světlo typu SUN.

Začněme parametrem random_spotlights. Pokud jsou světla zprovozňována s tímto parametrem, stanoví se seed a následně jsou vytvořeny čtyři světla typu SPOT. Každé z těchto světel je umístěno nad jeden z kvadrantů snímku a jejich energie je nastavena na hodnotu 25 pomocí proměnné spot_light_energy. Následně jsou všechna tato světla přidána do struktury list. Pro všechny z nich je provedeno následující nastavení. Jako první je hodnota parametru spot_blend nastavena na 1. Parametr spot_blend určuje měkkost hrany světelného kužele u těchto světel (Spotlight). Hodnota 1 znamená maximální měkkost hrany. Parametr spot_size je nastaven s využitím převodu stupňů na radiány na 50. Tím je určena šířka světelného kužele jako 50°. Parametr use_shadow při přepnutí na False opět provede vypnutí stínů. Poslední dva parametry se týkají samotného renderu. Při nastavení hide_viewport na True nebude světlo viditelné ve viewport renderu užívaném při anotaci. Nastavením hide_render je pak vypnuto i světlo pro render SEM snímku. Záhy potom, co jsou všechna světla takto nastavena se vybere náhodné (dle seedu) světlo z listu, a právě to je zapnuto nastavením hide_render na False. Výsledkem je tedy náhodně vybraný světelný zdroj ozařující jen určitou plochu z pozice nad vlákny. Tím vzniká světlá oblast ve snímku SEM. Takovéto oblasti je možné vidět i v reálných datech. Mohou být způsobeny různými faktory a v syntetických datech se je snažím simulovat právě tímto způsobem.

Přidání světla typu SUN je o dost jednodušší. Pokud je parametr zapnut, je v prvním kroku vytvořen světelný zdroj typu SUN. Jeho energie je na základě zkušeností nastavena na 3. Umístěno je zhruba do středu vláken a opět je nutné vypnout stíny. V druhém kroku je pak nastaven hide_viewport na True pro korektní anotaci. Toto světlo je do scény přidáváno, aby docházelo k jasnějšímu vyzáření okrajů vláken. Právě tak jsou totiž vyobrazena na reálných snímcích. Vlastnosti typu SUN jsou pro tento účel ideální. Na rozdíl od POINT má pro své paralelní paprsky lepší efekt. Díky nekonečném charakteru paprsků by pak světlo mělo osvítit všechna z vláken, na která dopadá, bez ztráty energie se vzdáleností. Zároveň by mělo světlo vyzařovat i pokud zrovna bude umístěno v objektu.

Konečně první parametr je nejdůležitějším. Jedná se o způsob nasvícení celé scény. Experimentováno bylo s více způsoby hlavního nasvícení. Prvním bylo nasvítit model z horní části (mód above). Tedy v podstatě z bodu, kde je kamera. Výsledkem byly vlákna s bílou střední částí a tmavými okraji. Vlákna si také nezachovávala dostatek trojrozměrnosti. Navazujícím způsobem bylo přidání světel ze všech stran (pokud se pozorovatel na model kouká z pohledu kamery) módem side_lights. Tedy z každé strany čtverce tvořícího půdorys obrázku. Takto nasvícená scéna získala o něco lepší trojrozměrný vzhled po vizuální stránce. Přes výsledky bližší realitě byla scéna poměrně přesvícená. V reakci na to jsem zachoval pouze světla na levé a pravé straně snímku. Tento typ obrázků nepůsobil tolik přesvíceně, a docházelo ke vzniku bílých hran jen u části vláken. Tuto část tvořila vlákna rovnoběžná, či skoro rovnoběžná s hranami, od kterých přicházelo světlo. Jako řešení jsem zvolil svícení na scénu čtyřmi světly, ovšem zkosením pod určitým úhlem. V prvním přístupu byla světla umístěna do horní poloviny strany části s vlákny (mód angled_side_upper). Při módu side_lights byl směr světla rovnoběžný s podložkou modelu. Po vychýlení se světla posunula na úroveň nejvyšších vláken a úhel se snížil o 10°. Tím vznikla světla svítící šikmo dolů ze stran scény. Výsledný snímek působil lépe než snímek z módu above. Zkoušel jsem i kombinaci pouze světel na levé a pravé straně. Na výsledek to ovšem nemělo znatelný pozitivní vliv. Dalším logickým krokem bylo použít obdobná světla, ovšem mířící nahoru. Tedy úhel sevřený s podstavou se zvětší. Na základě toho jsem vytvořil mód angled_side_lower. Výsledky takového nasvícení vypadaly více podobně reálným snímkům. Vlákna měla trojrozměrný vzhled, okraje byly světlejší a střed těla vlákna tmavší. Vzhledem k vizuální úspěšnosti byl pak vytvořen poslední z módů, a sice lower_fibers_only. Výstup tohoto módu byl vizuálně nejblíže reálným vláknům. To potvrdila i konzultace s pracovníky z KCH TUL. Přesné nastavení je možné vidět ve zdrojovém kódu 3.7. Obdobným způsobem, ale s jinými parametry, jako v tomto kódu, jsou nastaveny i všechny ostatní módy nasvícení.

```
right_light = create_light(name='right_light', energy=100, location_xyz=(8,4,0),
degree_rotation_xyz=(-7,78,7), area_shape='DISK')
right_light.hide_viewport = True
top_light = create_light(name='top_light', energy=100, location_xyz=(0,8,0),
degree_rotation_xyz=(-30,80,78), area_shape='DISK')
#top_light.hide_render = True
top_light.hide_viewport = True
left_light = create_light(name='left_light', energy=100, location_xyz=(-3,0,0),
degree_rotation_xyz=(168,107,12), area_shape='DISK')
left_light.hide_viewport = True
bottom_light = create_light(name='bottom_light', energy=100, location_xyz=(4,-
4,0), degree_rotation_xyz=(-23,78,-96), area_shape='DISK')
#bottom_light.hide_render = True
bottom_light.hide_viewport = True
above_scene_light = create_light(name='above_scene_light', energy=-5,
location_xyz=(2,2,5), shadow_enabled=False)
```

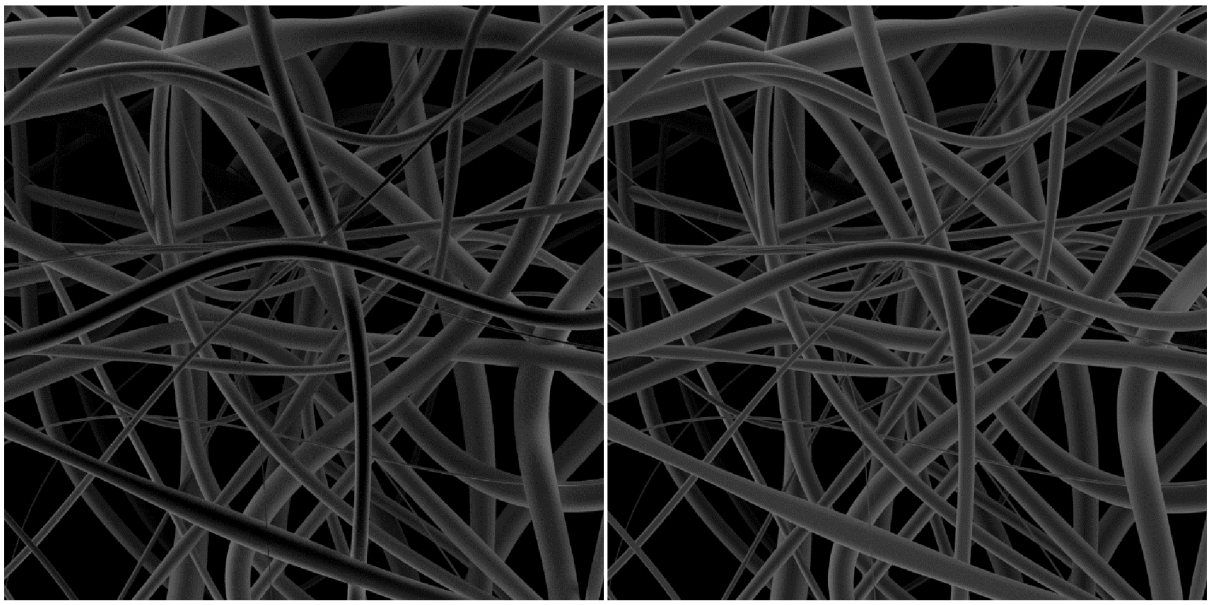
Zdrojový kód 3.7: Příklad nastavení světel funkcí apply_lightning_mode v módu lower_fibers_only

Kromě zmíněných způsobů nasvícení jsem se snažil také zahrnout do osvětlení fakt, že detektor je umístěn pouze na jedné straně. Tím pádem zde byla i myšlenka, že jas bude určitým způsobem orientován k jedné ze stran. V některých z reálných snímků lze tento jev lehce zpozorovat. Při pokusech rozšířit o jedno další světlo mířící ze strany jsem experimentoval

s různým typem světla, energií i úhlem. Žádnému ze snímků však tyto pokusy nepřidaly na realističnosti. Naopak snímek vypadal méně realisticky. Z toho důvodu nebyla tato idea v kódu realizována.

3.8 Volba render engine pro snímky SEM

V teoretické části jsem poukázal na fakt, že Blender obsahuje dva způsoby renderu obrázku. Těmi jsou EEVEE a Cycles. Z teorie plyne, že Cycles generuje obecně kvalitnější výstup za cenu větší náročnosti na hardware a časovou náročnost. Příkladem kvalitnějších komponent by mohli být např. stíny. Ty ovšem v projektu použity nejsou, a tak nehrají při výběru enginu roli. EEVEE naopak je obecně hlavně pro náhledy na výstup před finální render obrázku. Pro porovnání pochopitelně proběhl render oběma způsoby. Výstup je k vidění na obrázku 3.11.



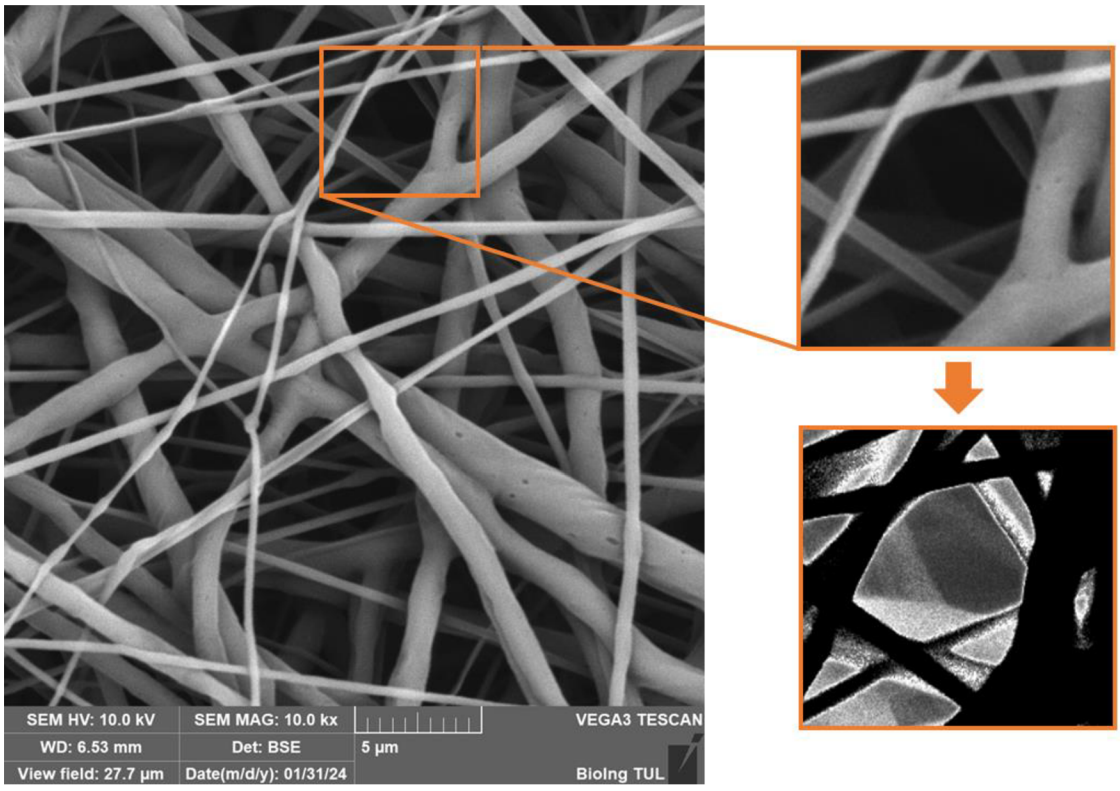
Obrázek 3.11: Na levé straně snímek renderování pomocí Cycles a na pravé pomocí EEVEE

Jako hlavní důvod volby EEVEE byl čas, který byl pro vytvoření snímku třeba. Výchozí hodnota vzorků renderování je 64 a čas na stroji, kde proběhl test všech typů engine byl dvě sekundy. Výstupní takto vzniklý snímek nejprve porovnejme se snímkem na pravé straně obrázku 3.11. Ten byl renderován také pomocí EEVEE a s 200 vzorky. Jeho tvorba pak trvala 5 sekund (vizuálně byl na stejné úrovni jako výchozí).

Oproti tomu výstup engine Cycles (pravá strana obrázku) měl při počtu vzorků 200 mnohem delší dobu pro render. Trval celé 3 minuty. Časová náročnost pro tvorbu našich dat nebyl prioritním požadavkem, ale přesto jsem upřednostnil rychlejší způsob poskytnutý EEVEE. Zvláště pak proto, že mezi výstupními snímky není kvalitativní rozdíl dostatečně velký, aby odůvodnil rozdíl časový.

3.9 Práce se snímkem mimo Blender

Reálné snímky obsahují vždy určité množství šumu. To, jak moc je obrázek zašuměný je závislé na způsobu skenování a rychlosti. Kvalitnější (méně zašuměné) snímky lze získat při rychlostech desítek až stovek mikrosekund na pixel. Zatímco větší míra zašumění vzniká v desetinách mikrosekund na pixel. Pro ilustraci šumu je uveden obrázek 3.12. Na obrázku je možné vidět reálný snímek nanovláken získaný pomocí SEM. Ve snímku je několik tmavších oblastí, které při pohledu na snímek vypadají jako čistě černé pozadí. Při změně barevného rozsahu však tyto oblasti odkryjí dva aspekty, které nejsou pouhým okem viditelné. Prvním je velmi tmavé vlákno. To pravděpodobně kvůli nedostatečnému rozlišení mikroskopu a upřednostnění kontrastu není vidět (jeho odstín splyne s pozadím). Druhým je šum. Tedy pixely mají různé hodnoty, ačkoliv na první pohled se zdají být jednotvárně černé.



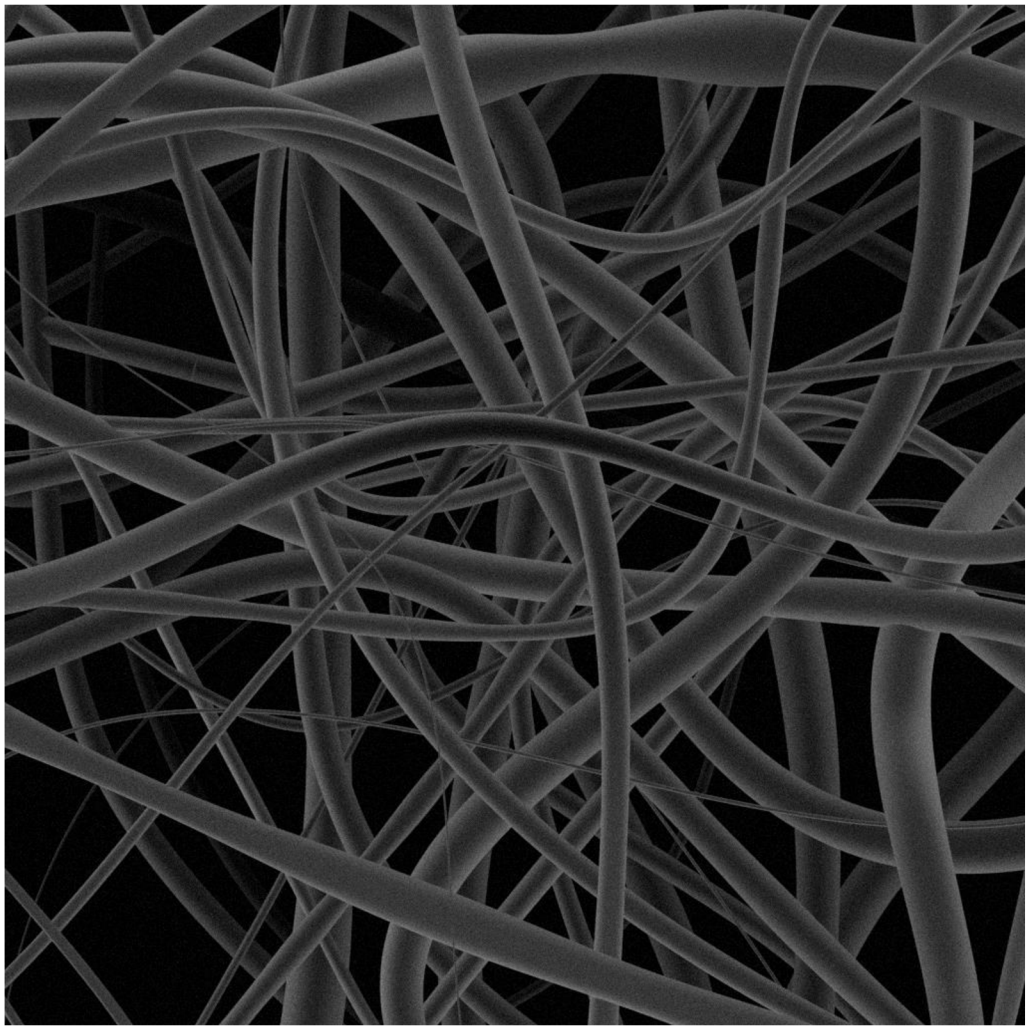
Obrázek 3.12: Šum v tmavých oblastech (pozadí) snímků SEM. Na levé straně je možné vidět původní snímek SEM, na pravé jeho výsek. Při posunutí hodnot pixelů jsou vidět vlákna na prvním snímku, která jsou okem nepozorovatelná, a šum

Vložení šumu do snímku by bylo třeba dělat přes materiály, jak je tomu v jednom z materiálů vláken. Pro vložení šumu na celý snímek jsem zvolil metodu pracující mimo Blender. K aplikaci šumu slouží funkce add_post_noise. Má tři parametry. Těmi jsou cesta ke snímku (image_path_name), na který má být šum aplikován, průměr (mean) a standartní odchylka (std_dev). Celá funkce je uvedená ve zdrojovém kódu 3.8 a pracuje hlavně s knihovnami OpenCV a NumPy.

```python
def add_post_noise(image_path_name, mean=0, std_dev=7):
    # Load the BW image
    image = cv2.imread(image_path_name, cv2.IMREAD_GRAYSCALE)

    # Add Gaussian noise
    gaussian_noise = np.random.normal(mean, std_dev, image.shape)
    noisy_image = np.clip(image + gaussian_noise, 0, 255).astype(np.uint8)

    # Save the noisy image, replacing the original (no compression)
    cv2.imwrite(image_path_name, noisy_image, [cv2.IMWRITE_PNG_COMPRESSION, 0])
```

Zdrojový kód 3.8: Popis funkce add_post_noise přidávající šum a pracující mimo Blender

Práce probíhá na snímku simulující SEM a standardně je tedy cesta nastavena na řetězec out/image{snimek_idx:03d}_SEM.png. To zajistí výběr předem generovaného snímku, kdy volání funkce je prováděno až po funkci sem_image_generating a save_img (k uložení snímku). Nejprve je vygenerovaný obrázek načten pomocí imread z knihovny z OpenCV formou grayscale. V dalším kroku je nastaven a připraven Gaussovský šum. Pomocí funkce np.random.normal jsou vytvořeny náhodné vzorky z normálního (Gaussova) rozdělení. Je tak učiněno na základě parametrů průměr, standartní odchylky a tvaru pole reprezentujícího obrázek (image.shape). Průměr určuje střední hodnotu šumu. Pokud je mean nulový, šum bude rovnoměrně rozložen kolem nuly. Pokud mean není nulový, šum bude posunut o tuto hodnotu. Standardní odchylka určuje rozptyl hodnot šumu kolem střední hodnoty. Čím větší je standardní odchylka, tím větší rozptyl hodnot šumu od střední hodnoty bude. Pokud je standardní odchylka nízká, šum bude mít tendenci být koncentrován kolem střední hodnoty s menší variabilitou. Naopak, pokud je standardní odchylka vysoká, šum bude mít větší variabilitu a rozptýlení hodnot kolem střední hodnoty bude větší. Jako výchozí hodnotu jsem zvolil 7. Pomocí funkce np.clip provedu součet vytvořeného pole šumu a původního obrázku. S tím, že určím spodní mez jako 0 a horní jako 255. Pokud by nějaká hodnota překročila tento rozsah, bude oříznuta na hodnotu 0 nebo 255 podle toho, zda je mimo dolní nebo horní mez. Nakonec tato metoda konvertuje všechny hodnoty pixelů výsledného obrazu na datový typ 8-bit unsigned integer (np.uint8). Nově vytvořené pole hodnot je uloženo jako obrázek s obdobným jménem jako byl obrázek původně načtený. K tomu slouží imwrite z OpenCV a pomocí jejího posledního parametru se zajistí nulová komprese. Příklad výsledného snímku je na obrázku 3.13.



Obrázek 3.13: Snímek simulující SEM s přidáním šumu mimo Blender

3.10 Výsledky dosavadního postupu a další možnosti vývoje

V první řadě bych rád zmínil přechod z původního 2D řešení do formy 3D. Toto rozhodnutí obnášelo opustit značnou část postupu a zároveň počátek učení se s kompletně novou sadou nástrojů. Nakonec se ovšem přechod na Blender stal poměrně velkým krokem vpřed. Jako posuny bych rád vyzdvihnul možnost lepší napodobení nasedání vláken na sebe (bližší reálnému výsledku zvláknění), poměrně elegantní způsob řešení interakce vláken (kolize atd.), způsob anotace na základě práce s materiálem v kombinaci s nastavením prostředí i světel a na závěr celkově lepší výsledky po vizuální stránce. Tím i lepší možnosti přiblížení se realitě.

Dále bych si dovolil zmínit nejdůležitější parametry, které je možné ovládat. Díky tomu si každý uživatel může vytvořit sadu snímků na základě svých požadavků. Dovolil jsem si to shrnout do tabulky 3.1. V tabulce jsou vždy uvedeny vizuální charakteristiky, které je možné zvolit pro generovanou sadu, parametr určující tuto vizuální složku a poznámky k prvkům. Tato tabulka rovněž slouží jako vodítko pro toho, kdo by chtěl v budoucnu na projekt navázat. Zároveň připomenu, že o prvcích týkající se vláken (především nastavení počtu, tloušťka, deformace na vláknech), je možné si podrobně přečíst v práci kolegy Davida Šafaříka.

Tabulka 3.1: Vizuální prvky nabízené pro generované snímky.

Vizuální prvek	Ovládání v kódu	Poznámky
Počet vláken	fiber_count	Napodobuje hustotu nanovlákenné vrstvy
Rozsah tloušťky vláken	radius_base_range	Pro ovládání homogenity tlouštěk vláken
Gamma	gamma_val	Ovlivňuje celkový odstín snímku
Nasvícení celé scény	apply_lightning_mode(), parametr mode	Možnost volby mezi více režimy (jako výchozí nasvícení vláken zespoda).
Náhodně umístěný světelný zdroj typu Spotlight	apply_lightning_mode(), parametr random_spotlights	Možnost zapnout nebo vypnout náhodné umístění světla typu Spotlight na jednu ze čtyř pozic
Zdroj světla typu Sun ve středu masy vláken	apply_lightning_mode(), parametr add_center_sun	Poskytuje větší zvýraznění světlých okrajů vláken
Přerušení vlákna	number_of_fibers_cut_in_middle	Počet vláken s touto deformací
Kapková deformace na vlákně	number_of_fibers_with_droplet	Počet vláken s touto deformací
Rozdvojení vlákna	number_of_fibers_with_Y_division	Počet vláken s touto deformací
Vložení nevlákenného objektu	add_objects()	Pro simulaci nečistot a dalších objektů vyskytujících se ve snímcích
Standartní materiál	sem_image_generating	Materiál bez šumu barvou závislou na hloubce umístění vlákna
Experimentální materiál (s šumem)	sem_image_generating(), parametr use_noise_tex	Materiál s texturou šumu tvořenou přímo v Blenderu (ovlivňuje barvu vláken)
Zanesení šumu do snímku	post_noise	Po kompletním zpracování je zanesen (Gaussovský) šum na celý snímek

3.10.1 Informace o síti a úvod do způsobu vyhodnocení

Výstupní data byla uspořádána do sad, na kterých následně probíhalo učení. V každé sadě jsem se snažil zachytit různé prvky z reálných snímků SEM. Cílem pak bylo, data s těmito prvky předat síti a sledovat, jak napomáhají kvalitnějšímu rozpoznání. A selektovat ty prvky, které zdánlivě vedou k nejlepším výsledkům. Samotné rozpoznávání bylo řízeno ze strany pana doktora Pavla Mártona. Jemu jsme vždy předali, na základě jeho požadavků, sadu pěti snímků s anotacemi. Každou sadu tvořili tři snímky trénovací a dva validační.

Síť samotná vznikla dle pana vedoucího na základě práce Ronneberger et al. [34] a architektuře zde popsané. V síti je používáno 8 konvolučních jader na první úrovni a dvojnásobný počet jader při každém přechodu do nižší úrovně. Jako aktivační funkce je použita ReLU (Reclified Linear Unit) a pro redukci dat při přechodu do nižší úrovně sítě maxpooling o rozměru 2x2. Na nejnižší úrovni pak není prováděn maxpooling. Ve vzestupné části sítě typu U-Net je užito dilatace o rozměru 2x2, spojení s daty stejného rozměru z klesající části a postupné snižování počtu konvolučních jader (pro danou úroveň stejný počet jako při cestě k nižším úrovním).

Pro účel ověření, jak si natrénovaná síť povede na reálných obrázcích, jsme vybrali pět snímků skutečných nanovláken (vzorky uvedeny v příloze). Na těchto snímcích jsem pomocí ImageJ naměřil sto hodnot průměrů. Měřena byla různá vlákna, ale i různé části stejných vláken v celé ploše snímku. U těchto snímků jsme zaznamenali středy úsečky (souřadnice x, y), její délku a úhel. Na základě těchto tří údajů jsme pak replikovali řezy v našich rozpoznaných datech. Ve výsledku jsme porovnávali v rámci jednoho řezu délku původní úsečky a hodnoty pixelů ve výstupním snímku sítě. V ideálním případě by hodnota všech pixelů byla rovna délce úsečky z originálních snímků. Vzhledem k tomu, že výstupní obrázky byly formátu 512x512, zatímco originální snímky jsou ve formátu 1024x1024, bylo třeba sjednotit rozměry. Za tímto účelem jsme pomocí nearest neighbor interpolace zvětšili výstupní snímky sítě.

3.10.2 Hodnocené sady snímků, jejich příprava a výsledky hodnocení

Pro určení ukazatele přesnosti, jak dobře síť rozpoznává průměry, jsem vypočítal normu rozdílů anotovaného snímku a snímku s odhadem sítě. Tím byla přesnost popsána jedním číslem a norma byla použita jako ukazatel úspěšnosti. Vyšší číslo naznačovalo větší rozdílnost a menší naopak větší podobnost. Sledováno bylo rozpoznávání na našich datech (trénovací a validační) a reálných datech (anotovaných ImageJ). V prvním případě jsem provedl tuto analýzu na našich datech.

V případě našich dat byla v každém snímku vypočtena norma rozdílu anotovaného snímku a odhadu sítě. Následně byl spočítán průměr pro každou sadu. Konkrétně průměr přes trénovací data a následně přes validační. Prvním náznakem úspěchu při učení sítě bylo porovnání trénovacích a validačních dat. Ukazatel úspěšnosti byl totiž ve všech třídách nižší pro trénovací než pro validační. Zároveň i v samotných trénovacích datech nebyl ukazatel nulový. To znamenalo, že síť nebyla přetrénována. Rozdíl mezi oběma částmi sady nikdy nebyl příliš velký, což také poukázalo na nepřetrénovanost. V další části se věnuji porovnáváním výsledků jednotlivých sad. Je třeba říct, že v rámci námi tvořených snímků vycházely jako lépe rozpoznané jiné sady než v reálných snímcích. Je však třeba zmínit, že hodnoty ukazatele byly i tak poměrně velké na to, že se jednalo o kontrolovaná data. Ta by měla být v ideálním případě

menší. Pochopitelně kromě samotných snímků hraje velkou roli i funkčnost sítě. To už ale není předmětem této práce.

U reálných dat, jak bylo popsáno, se pracovalo v rámci lokálních úseček. Zde byl změřen průměr a pak porovnán s odhadem sítě. Opět se provedl rozdíl a z něj vypočetla norma. Tím byl získán ukazatel úspěšnosti určení průměru vlákna. V každém ze snímku byly pro jednotlivých sto měření vypočteny normy a ty zprůměrovány. Výsledky byly pochopitelně ovlivněny tím, jak vypadala trénovací data konkrétní sady. V sadách bylo pochopitelně několik faktorů, které výsledek ovlivnily. Výsledky pro jednotlivé sady jsou uvedeny v tabulce 3.2.

Tabulka 3.2: Ukazatelé úspěšnosti pro jednotlivé sady (měření na reálných SEM snímcích)

Označení	**Prvky ve snímku**	**Ukazatel úspěšnosti**
A	Světlo umístěno nahoře, gamma = 1	80,32449254
B	Světlo side_lights + SUN uprostřed, gamma = 1	68,56418619
C	Světlo lower_fibers_only + SUN uprostřed, gamma = 1	140,2431354
D	Světlo lower_fibers_only + SUN uprostřed + Spotlight, gamma = 1	93,18622913
E	Světlo lower_fibers_only + SUN uprostřed + Spotlight, gamma = 1, materiál s texturou šumu	83,02689038
F	Světlo lower_fibers_only + SUN uprostřed, gamma = 2	116,9725428
G	Světlo lower_fibers_only + SUN uprostřed, gamma = 2, materiál s texturou šumu	90,97728185
H	Obdobně jako F, ale s lepší anotací průměrů	87,22461305
I	Obdobně jako G, ale s lepší anotací průměrů	70,78507384
J	Světlo lower_fibers_only + SUN uprostřed, gamma = 1, šum přes celý obrázek	147,9744016
K	Světlo lower_fibers_only + SUN uprostřed, gamma = 2, šum přes celý obrázek	106,9135155
L	Světlo lower_fibers_onl' + SUN uprostřed, gamma = 1, materiál s texturou šumu, šum přes celý obrázek	88,5817659
M	Světlo lower_fibers_only + SUN uprostřed, gamma = 1, materiál s texturou šumu, počet vláken 100	84,1532621
N	Světlo lower_fibers_only + SUN uprostřed, gamma = 1, materiál s texturou šumu, počet vláken 400, základní poloměr v rozsahu 0,003 až 0,03	75,52949803
O	Světlo lower_fibers_only + SUN uprostřed, gamma = 1, počet vláken 400, základní poloměr v rozsahu 0,003 až 0,03	77,58412875
P	Světlo side_lights + SUN uprostřed + Spotlight, gamma = 1, počet vláken 100 (na základě sady B)	75,33294504

V prvním kole byly řešeny sady A až L. U všech těchto sad byl počet vláken 50. Zbylé parametry sad jsou uvedeny v předchozí tabulce. Z vizuálního hlediska si nejlépe vedly sady

označené B a I. Pro sadu B (příklad v B.1) bylo využito nasvícení ze stran, gamma nastavena na hodnotu 1 a užit zdroj světla typu SUN uprostřed. Po vizuální stránce se jednalo o překvapivý výsledek. Nasvícení ze stran se na základě konzultací s lidmi z oboru zdálo jako méně podobné realitě. I přes to si tato sada vedla lépe než zbylá nasvícení. Paradoxně právě první tři sady (A, B, C) byly zaměřeny na nasvícení. Přitom si sada s osvícením spodních vláken vedla hůře než zbylé dvé. A to i přesto, že byla shledána na konzultacích jako nejpodobnější reálným datům. V sadě s nejlepším výsledkem je zahrnut i zdroj světla SUN, který přispívá k osvícení krajů. Ten na základě výsledků k úspěšnosti přispívá. Co se samotných podobností zachycených ve snímku týče, jednalo se dle mého názoru především o dobré rozložení světlých a tmavých částí vláken.

Druhou úspěšnou sadou byla I. Zde se jednalo o kompletně jiný přístup než u sady B. Nasvícení bylo směřováno na spodní vlákna. Dalším rozdílem bylo použití materiálu s texturou šumu. Hodnota gamma byla 2. Snímky z této sady vynikaly dosti světlým vizuálem, který by mohl být příčinou dobrého rozpoznání. Abych pak navázal na téma jasu snímku, tak porovnáme-li snímky J a K, je mezi nimi jen upravena hodnota gamma. Vyšší hodnota 2 vede k o dost lepšímu výsledku rozpoznávání. Nebýt tak dobrého výsledku předchozí sady B, dalo by se vyvozovat, že vyšší jas snímku (využita větší gamma) jednoznačně k lepšímu výsledku vede. I přes to bych doporučil ho při tvorbě budoucích sad doporučil užívat ve vyšší hodnotě.

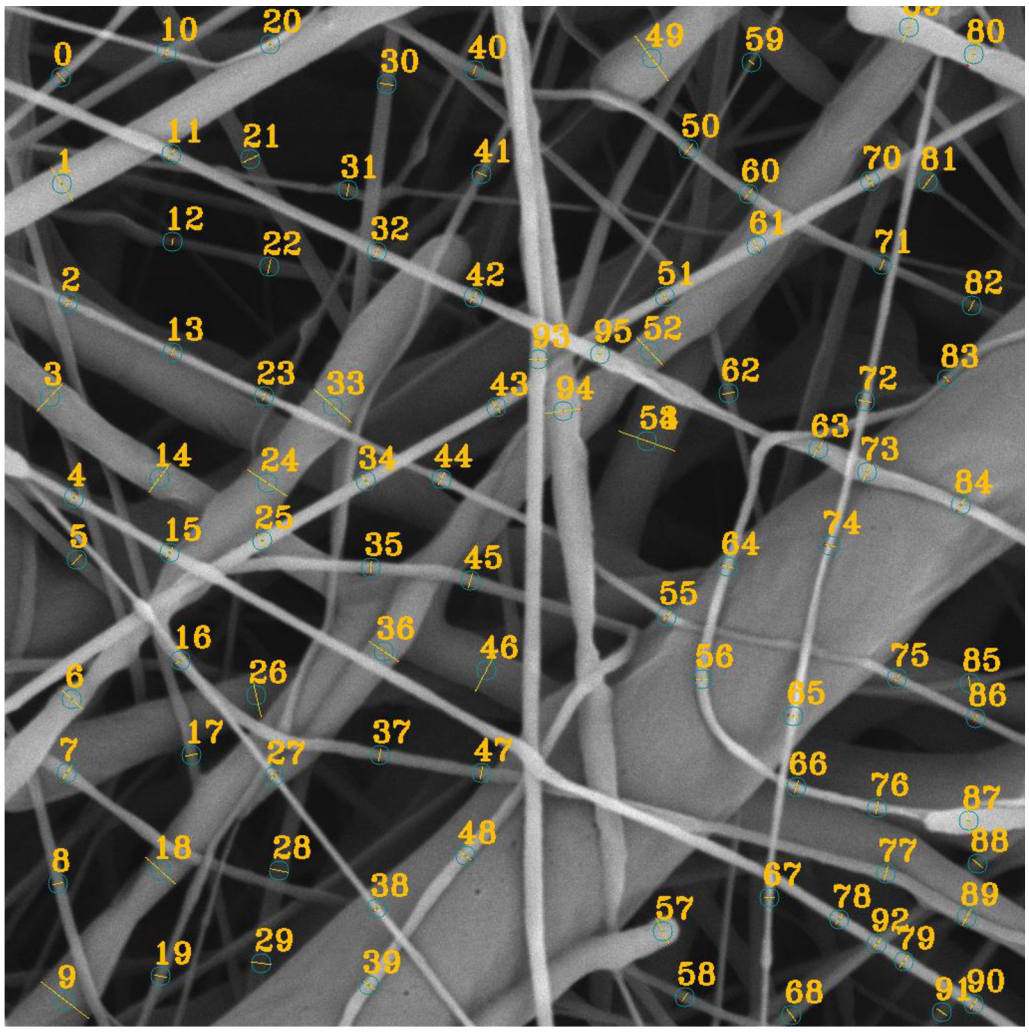
Zaměříme-li se na šum, který je přítomen v reálných snímcích. Rámci programu je možné ho nanést jak na materiál vláken, tak na celý snímek (pomocí různých postupů). Nejprve se podívejme na sady D a E. Obě sady jsou totožné až na materiál. Sada E s materiálem šumu má mnohem lepší výsledky. Opět nebýt nejúspěšnější sady B, by bylo možné vyvodit jednoznačně pozitivní vliv, i pro použití šumu přes celý snímek (sady J, K, L). V tom případě dochází ke značnému snížení úspěšnosti rozpoznávání. Tento jev je zvláštní, protože jak bylo doloženo v předchozí částí práce, šum je přítomný ve všech reálných snímcích. Zároveň je ve zmíněné trojici sad vidět, že nejlepší z nich je právě ta s použitou texturou šumu na materiálu vlákna. Až na rovnoměrnější rozdělení světlých a tmavých částí, které jsou na reálných snímcích SEM, je obdobně pozvolnější přechod mezi tmavým středem a světlými okraji. Stejně jako tomu je v reálných snímcích SEM i nejúspěšnější sadě B.

V reálných snímcích se lze často setkat s oblastmi s vysokým kontrastem. Většinou se tyto oblasti vyjímají více světlejší než okolí. Ty jsem se snažil do snímku zanést užitím Spotlight. Obsahují je sady D a E. Nejedná se sice o sady s největší úspěšností, nicméně i tak mají poměrně dobrý výsledek. Připustíme-li další ovlivnění tohoto výsledku jinými parametry, pak je pravděpodobné, že Spotlight napomohl při učení sítě k lepšímu natrénování.

Druhou sérii sad tvořily sady M až P. Zde jsem se zaměřil především na zdůraznění úzkých vláken, počet vláken a snahu o vylepšení nejlepších sad z první skupiny. Větší počet úzkých vláken v sadách N a O pravděpodobně poskytl síti lepší materiál pro trénování. Obě sady si vedly velmi dobře. Zároveň sada N byla ještě obohacena o texturu s šumem na materiálu. V porovnání obou sad pak vyšla lépe, což bych připsal právě textuře. Na závěr sada P byla v podstatě pokus o vylepšení sady B. Ačkoliv si vedla dobře, tak i přes doplnění prvků, které jiným sadám napomohly, nedosáhla výsledků jako sada B. Tento nedostatek jí nejspíš ubral materiál nebo světlo typu Spotlight.

3.10.3 Lokální analýza na vláknech

Na obrázku 3.14 je jeden ze pěti snímků anotovaných ImageJ. Jsou na něm zvýrazněny úsečky s čísly užité v měření. Jeden z konců úsečky se vyznačuje červenou šipkou. Právě ten je jejím počátečním bodem. Měření v tomto snímku byla provedena na úsecích vláken, kde by mělo být rozpoznání pro síť nejsnazší. Z těchto měření byla následně vypočtena hodnota určující kvalitu určení průměrů sítí. Snímky, rozpoznané ze sítí, jsem pro úspornost do práce neuváděl. Budou zde jen uvedeny jejich konkrétní výřezy s komentářem (viz níže). Nutno dodat, že výstupem byly i úsečky v místech, kde bylo rozpoznávání nějakým způsobem ztíženo (např. překrytím dvou vláken). Tyto případy byly řešeny samostatně. U rozpoznaných snímků se lze při vizuální kontrole řídit tím, že čím je barva tmavší, tím užší je vlákno (stejně jako v anotaci syntetických). Pro širší vlákna to platí naopak. V ideálním případě by mělo být pozadí zcela černé (hodnota 0) a jednotlivá vlákna odstupňované právě dle hodnoty šířky.



Obrázek 3.14: Snímek se zakreslenými úsečkami na základě ImageJ (vzorek A.1 ES_Pavla_PCL-dmf+AuNP_10kx_01xxBSE_true)

Pro lokální analýzu jsem zvolil čtyři dvojice (značné A, B, C, D). Dvojici tvoří vždy výsek z rozpoznaného snímku a graf zachycující, jak se liší hodnoty pixelů od reálné hodnoty průměru. Každá nese pro přehlednost označení a všechny jsou ze sady B, která si v prvním

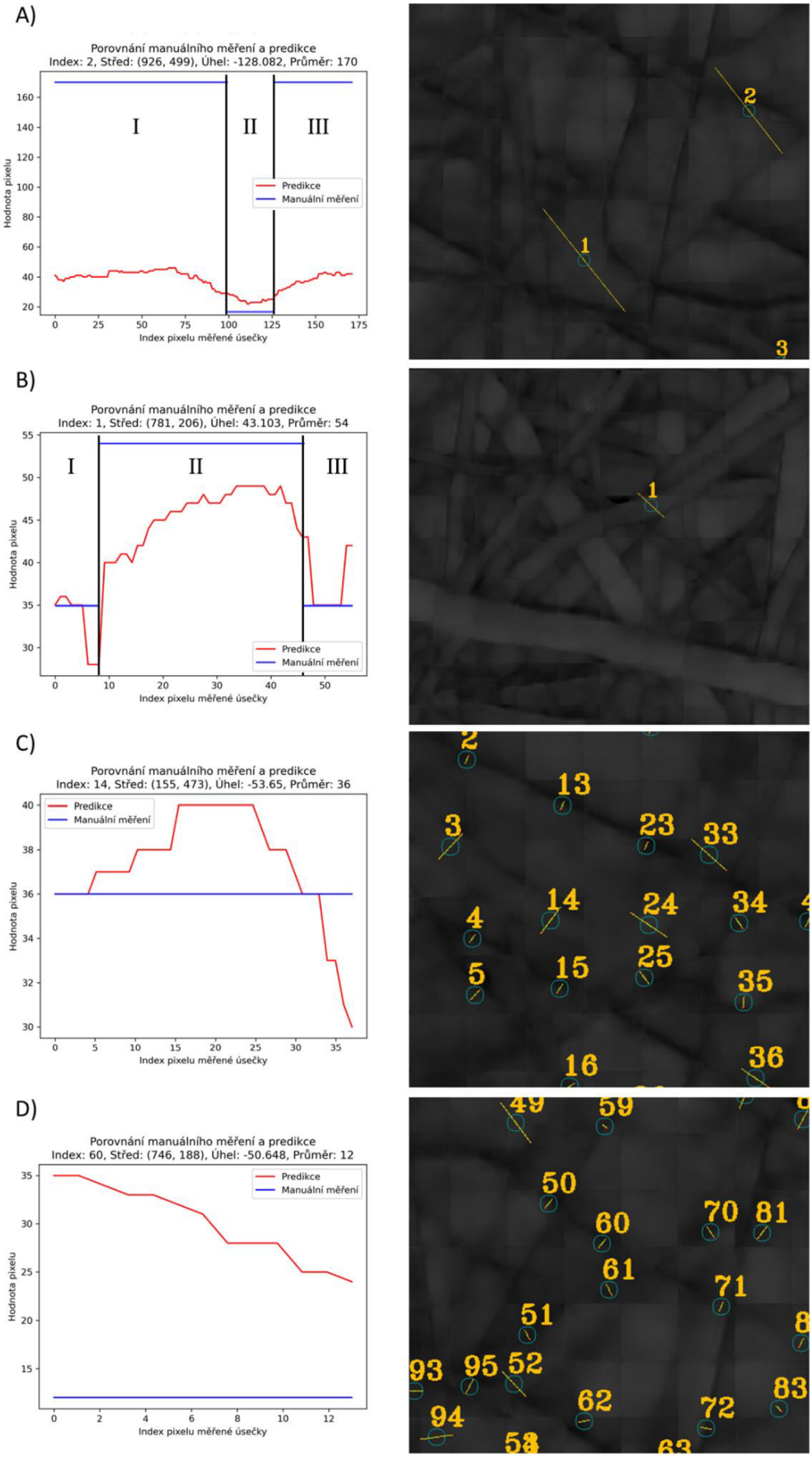
kole vedla nejlépe. Grafy jsou obohaceny o pomocné vysvětlující prvky. Sjednotil jsem je do jednoho obrázku, který je možné najít pod číslem 3.15. Následuje detailní popis tří vzorových případů.

V případě A se jedná o široké vlákno překryté úzkým vláknem (měření s indexem 2). Je to vlastně výřez ze vzorku ES_Pavla_PCL-dmf+AuNP_10kx_01xxBSE (A.1). Tento případ je velice specifický díky velkému rozdílu obou figurujících vláken. Provedl jsem měření průměrů obou vláken a také částí, které jim na úsečce náleží. Silnější vlákno má rozměr 170 pixelů a tenčí 18 pixelů. V případě úzkého vlákna bylo měřeno místo, kde došlo k přetnutí. Celou úsečku lze pak rozdělit na tři části (I, II, III). V ideálním případě by dvě krajní části (I a III) nesly hodnotu 170 a prostřední (II) 18 pixelů. První věc, kterou je možné v grafu pozorovat, je snížení v oblasti úzkého vlákna. To je samo o sobě úspěch potvrzující rozlišení dvou vláken. Bohužel rozlišení není ideální, jinak by byl přechod mnohem strmější, nikoliv takto pozvolný. Další fakt je, že místo s úzkým vláknem je mnohem blíže reálné hodnotě než širší vlákno. Síť si tedy nebyla schopna přesně poradit s takto širokým vláknem. Na druhou stranu vlákno užší, rozpoznala poměrně dobře. To je zajímavým faktem, protože úzká vlákna jsem jedním z větších nedostatků dat.

Případě B je opět dvojice vláken, kdy jedno poměrně plynule sestupuje pod druhé (měření s indexem 1). Jedná se o vzorek PCL80-5kx (A.4). Vrchní vlákno (sekce II) má průměr 54 pixelů a vlákno ve spodní vrstvě (sekce I a III) 35 pixelů. V tomto případě jsou vlákna poměrně razantně odlišena, což se podepisuje na strmém sestupu v grafu. Zároveň v první sekci je značná část predikce směřována k ještě nižším hodnotám. To může být způsobeno tím, že síť odhaduje okraj jako úzké vlákno (zmíněno dříve). Stejně jako v předchozím příkladu se zde setkáváme s podhodnocením průměru širších vláken.

Případ C je označen indexem 14, vzorek A.1. Tento řez jsem zvolil, protože vlákno je poměrně jasně odděleno od pozadí, má jednu z běžnějších šířek a obecně na něm nejsou žádné náročnější prvky (překrytí atd.). Jak je vidět, odhad sítě zde převyšuje reálný průměr. Tím se liší od předchozích dvou vláken. Počátek úsečky a následně trend stoupání síť vystihla přesně. Zajímavý je strmý pokles na koncové časti měřené úsečky. Zde je opět vidět pokles interpretovaný jako užší vlákno. Nicméně na originálním snímku není okraj výrazněji světlý. Je ale možné, že tento jev způsobil přechod na tmavé pozadí, které se zde nachází. Znamenalo by to nejspíš, že sada neposkytovala dostatečnou informaci pro rozpoznání pozadí od vlákna.

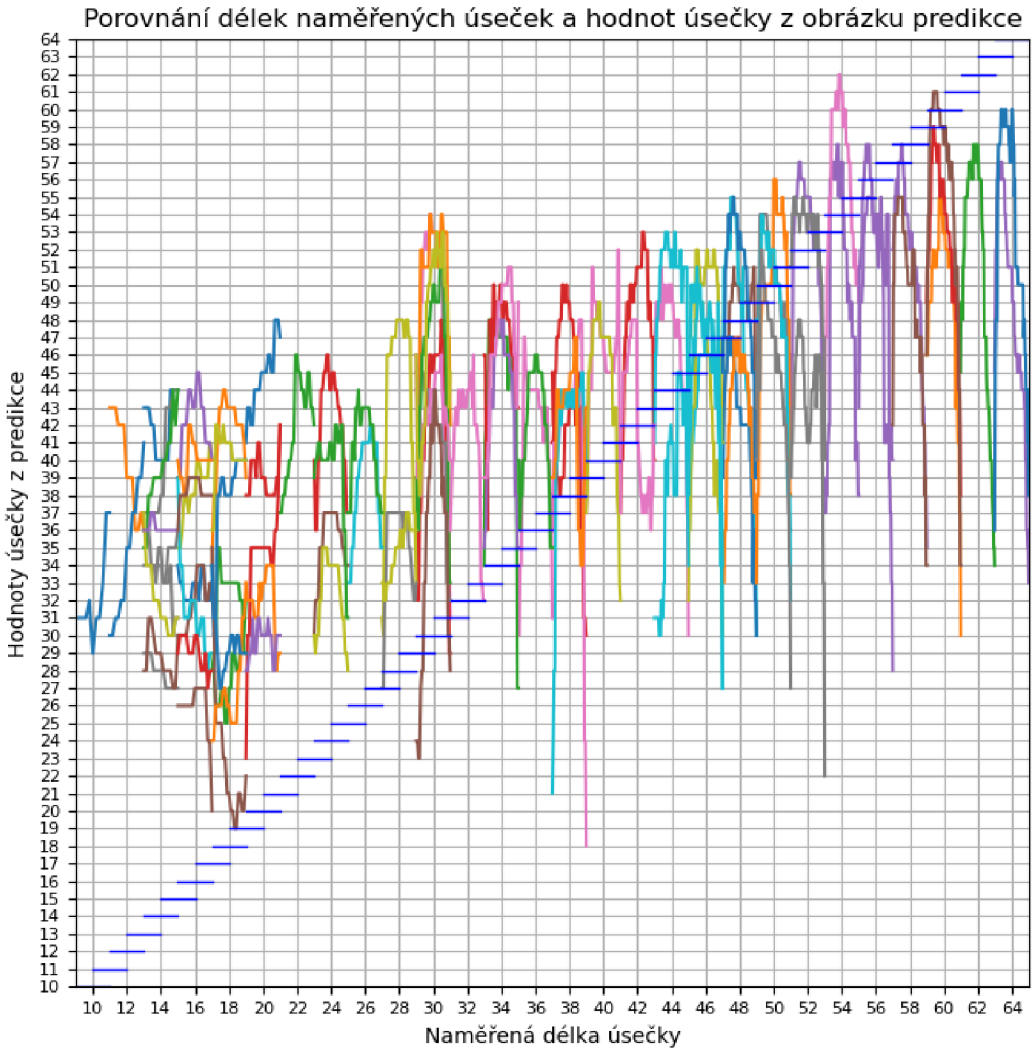
Případ D je označen indexem 60 a je opět v již zmíněném vzorku (A.1). Toto vlákno je jedno z vláken s menším průměrem. Odhadovaná velikost je tedy větší než v reálu. To je jev, který bylo možné sledovat i v případu A (konkrétně sekce grafu II). V tomto případě je viditelné ještě minimálně jedno další vlákno, jež prochází pod vláknem měřeným. Nelze potvrdit, zda i toto vlákno nějakým způsobem neovlivnilo měření. Sestup odhadovaných hodnot je o dost plynulejší než v předchozích případech. Vlákno má v originálu poměrně uniformní odstín bez jasnějších okrajů. Na závěr, z pohledu vizualizace, celková odchylka odhadovaná od reálné hodnoty, by mohla být opět způsobena nižším zastoupením užších vláken v sadách.



Obrázek 3.15: Grafy a výřezy snímků pro lokální analýzu

3.10.4 Hodnocení přes různorodá vlákna

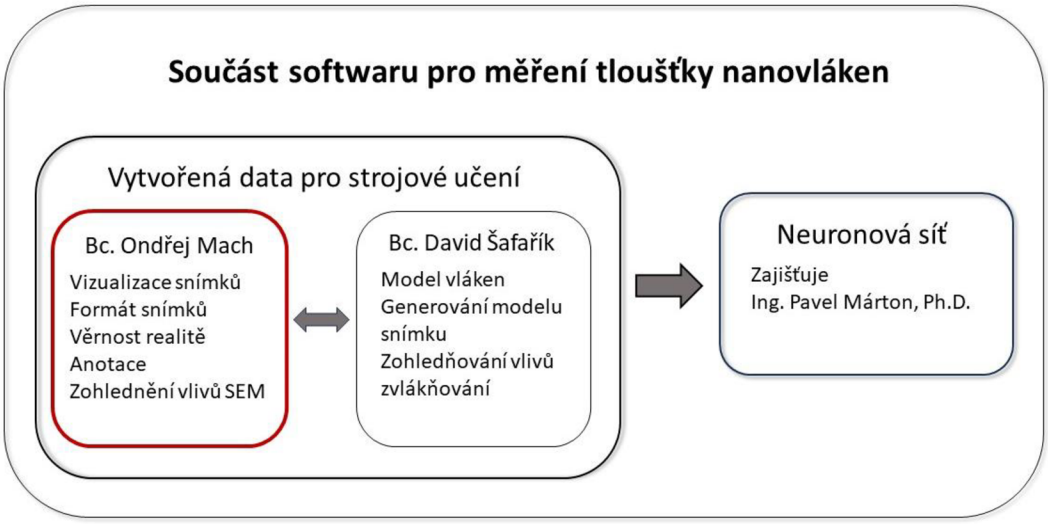
V obrázku 3.16 jsou znázorněna veškerá lokální měření jakožto celek. Jednotlivé řezy reprezentující průměry vlákna jsou seřazeny. V ideálním případě by se tak jednalo o sérii úseček s lineárně rostoucím umístění v grafu (v grafu znázorněno pomocí modrých linek). Tak tomu bohužel není. I přesto je možné z grafu vyčíst následující. Na základě hodnot síť nejspíše lépe rozeznává širší vlákna než ty užší. Dále nedochází ke korektnímu rozeznávání okrajů vláken. Ta jsou možná sítí identifikována jako samostatná a úzká vlákna. Nejlépe jsou asi rozeznána vlákna s hodnotou průměru mezi 30 a 40 pixely. Těmto problémům by se z hlediska dat dalo pravděpodobně lépe předcházet užitím většího počtu úzkých vláken, jak je vidět při generování sad v druhé sérii. Případně tvorbou větší sady, kde budou tvořit část právě takové snímky. Obecně bych si dovolil navrhnout tvorbu větší sady, do které by bylo možné zahrnout více druhů snímků.



Obrázek 3.16: Obrázek grafu se závislostmi naměřené délky a hodnot predikce. Jedná se o všechny seřazené hodnoty ze sady B

4 Závěr

Hlavním cílem práce bylo přispět k vývoji programu pro generování snímků simulující obrázky nanovláken ze SEM. Dílčími kroky byla rešerše SEM, anotace tvořených dat a určení formátu snímků specifikovaných v zadání (ve spolupráci s Davidem Šafaříkem). To vše s cílem o dosažení co nejvěrnějších dat realitě. Práce se věnuje především vizuální stránce a oblasti SEM. Výstupní program by měl tvořit data pro strojové učení. Díky tomu by měl vzniknout nástroj pro měření průměru nanovláken. Samotné rozložení projektu je uvedeno na obrázku 4.1.



Obrázek 4.1: Rozložení projektu

Po přechodu na užívání nástroj Blender byl vyvinut nástroj, který zajišťuje tvorbu snímků ve formátu specifikovaném v zadání. Společně s ním vznikl i způsob anotace těchto snímků. Primárně anotací průměru, nicméně byly zprostředkovány i další dílčí anotace. Pro podobnost dat bylo využito různých nástrojů poskytovaných ve vývojovém prostředí. Těmi byly např. ovládání světel, práce s materiálem a také zpracování mimo Blender. Díky tomu vzniklo několik reprezentativních sad, jejichž kvalita byla testována při trénování neuronové sítě. Celý program je také řešen tak, aby bylo možné s co největší volností nastavovat parametry pro tvorbu sad se specifickými potřebami. Zároveň je celý skript rozsáhle okomentován a je tedy vhodně připraven pro další možný vývoj.

Samotné sady byly generovány v podstatě ve dvou větších sériích (podrobněji v předchozí kapitole). První série sad byla více experimentální, zatímco druhá se zaměřovala na nedostatky a rozšiřovala sérii úvodní. Úspěšnost jednotlivých sad jsme hodnotili na základě ukazatele úspěšnosti. Ten je také podrobně popsán v poslední části praktické části. Dovolím si jen zmínit, že nejlepší výsledky vykázala sada s označením B (jako příklad uvedena v přílohách). Ta se konkrétně vyznačovala užitím osvětlení side_lights + SUN, hodnotou gamma rovnou jedné, standartním materiálem a počtem 50 vláken. Zde bych také doporučil pro další vývoj začít. Vliv různých parametrů je také rozebrán v předchozí části.

Pro shrnutí bylo tedy výsledkem práce příspěvek do programu pro generování snímků a jejich náležitostí. Vše odpovídá zadání a bylo důkladně popsáno jednak přímo v tomto textu, ale také v hojných komentářích zahrnutých v kódu.

Dovolím si tvrdit, že program může používat pro generování i člověk s nižší úrovní znalosti programování. Vzhledem k tomu, že se jednalo o inovativní přístup, bylo v průběhu vývoje vytvořeno nemalé množství objevů této tématiky. K tomu patřila i volba a rozvoj programu v 3D prostředí. Ačkoliv byl připraven solidní základ a bylo dosaženo určitých výsledků, tak je zde další prostor pro další vývoj, a to v oblasti dat, ale i sítě.

Do budoucího vývoje bych doporučil především rozšířit množství volitelných abnormalit, práci na materiálu a případně možnost další anotace, kterou je anotace křivosti. Ta by v budoucnu mohla být na základě konzultace s odborníky na nanomateriály také zajímavá.

Použitá literatura

[1] LUKÁŠ, David. Nanovlákna: teorie, technologie a použití. první. Praha: Academia, 2023. ISBN 978-80-200-3400-7.

[2] UL-HAMID, Anwar. A Beginners' Guide to Scanning Electron Microscopy [online]. Cham: Springer International Publishing, 2018 [vid. 2024-03-29]. ISBN 978-3-319-98481-0. Dostupné z: doi:10.1007/978-3-319-98482-7

[3] WILSON, A. The formation of dry, wet, spunlaid and other types of nonwovens. In: Applications of Nonwovens in Technical Textiles [online]. B.m.: Elsevier, 2010 [vid. 2024-03-17], s. 3–17. ISBN 978-1-84569-437-1. Dostupné z: doi:10.1533/9781845699741.1.3

[4] ALGHORAIBI, Ibrahim. Different methods for nanofibers design and fabrication. In: [online]. 2018. ISBN 978-3-319-42789-8. Dostupné z: doi:10.1007/978-3-319-42789-8_11-2

[5] BARHOUM, Ahmed, Kaushik PAL, Hubert RAHIER, Hasan ULUDAG, Ick Soo KIM a Mikhael BECHELANY. Nanofibers as new-generation materials: From spinning and nano-spinning fabrication techniques to emerging applications. Applied Materials Today [online]. 2019, **17**, 1–35. ISSN 2352-9407. Dostupné z: doi:10.1016/j.apmt.2019.06.015

[6] NAYAK, Rajkishore, Rajiv PADHYE, Illias Louis KYRATZIS, Yen Bach TRUONG a Lyndon ARNOLD. Recent advances in nanofibre fabrication techniques. Textile Research Journal [online]. 2012, **82**(2), 129–147. ISSN 0040-5175, 1746-7748. Dostupné z: doi:10.1177/0040517511424524

[7] VASS, Panna, Edina SZABÓ, András DOMOKOS, Edit HIRSCH, Dorián GALATA, Balázs FARKAS, Balázs DÉMUTH, Sune K. ANDERSEN, Tamás VIGH, Geert VERRECK, György MAROSI a Zsombor K. NAGY. Scale-up of electrospinning technology: Applications in the pharmaceutical industry. WIREs Nanomedicine and Nanobiotechnology [online]. 2020, **12**(4), e1611. ISSN 1939-0041. Dostupné z: doi:10.1002/wnan.1611

[8] SARABI-MIANEJI, Siavash, Jennifer SCOTT a Danny J.Y.S. PAGÉ. Impact of electrospinning process parameters on the measured current and fiber diameter. Polymer Engineering & Science [online]. 2015, **55**(11), 2576–2582. ISSN 1548-2634. Dostupné z: doi:10.1002/pen.24150

[9] EBERLE, A.l., S. MIKULA, R. SCHALEK, J. LICHTMAN, M.l. Knothe TATE a D. ZEIDLER. High-resolution, high-throughput imaging with a multibeam scanning electron microscope. Journal of Microscopy [online]. 2015, **259**(2), 114–120. ISSN 1365-2818. Dostupné z: doi:10.1111/jmi.12224

[10] ZHOU, Weilie, Robert APKARIAN, Zhong Lin WANG a David JOY. Fundamentals of Scanning Electron Microscopy (SEM). In: Weilie ZHOU a Zhong Lin WANG, ed. Scanning Microscopy for Nanotechnology: Techniques and Applications [online]. New York, NY: Springer, 2007 [vid. 2024-03-17], s. 1–40. ISBN 978-0-387-39620-0. Dostupné z: doi:10.1007/978-0-387-39620-0_1

[11] Which Electron Detector is Right for Your Application? | Nanoscience Instruments [online]. [vid. 2024-03-30]. Dostupné z: https://www.nanoscience.com/blogs/which-electron-detector-is-right-for-your-application/

[12] CAZAUX, Jacques. From the physics of secondary electron emission to image contrasts in scanning electron microscopy†. Journal of Electron Microscopy [online]. 2012, **61**(5), 261–284. ISSN 0022-0744. Dostupné z: doi:10.1093/jmicro/dfs048

[13] ImageJ Wiki. ImageJ Wiki [online]. [vid. 2024-03-31]. Dostupné z: https://imagej.github.io/

[14] SCHNEIDER, Caroline A., Wayne S. RASBAND a Kevin W. ELICEIRI. NIH Image to ImageJ: 25 years of image analysis. Nature Methods [online]. 2012, **9**(7), 671–675. ISSN 1548-7105. Dostupné z: doi:10.1038/nmeth.2089

[15] HOTALING, Nathan A., Kapil BHARTI, Haydn KRIEL a Carl G. SIMON. DiameterJ: A validated open source nanofiber diameter measurement tool. Biomaterials [online]. 2015, **61**, 327–338. ISSN 0142-9612. Dostupné z: doi:10.1016/j.biomaterials.2015.05.015

[16] MURPHY, Ryan, Ashley TURCOTT, Leo BANUELOS, Evan DOWEY, Benjamin GOODWIN a Kristen CARDINAL. SIMPoly: A Matlab-based Image Analysis Tool to Measure Electrospun Polymer Scaffold Fiber Diameter. Tissue Engineering Part C: Methods [online]. 2020, **26**. Dostupné z: doi:10.1089/ten.TEC.2020.0304

[17] GÖTZ, Andreas, Volkmar SENZ, Wolfram SCHMIDT, Jennifer HULING, Niels GRABOW a Sabine ILLNER. General image fiber tool: A concept for automated evaluation of fiber diameters in SEM images. Measurement [online]. 2021, **177**, 109265. ISSN 0263-2241. Dostupné z: doi:10.1016/j.measurement.2021.109265

[18] HULING, Jennifer, Andreas GÖTZ, Niels GRABOW a Sabine ILLNER. GIFT: An ImageJ macro for automated fiber diameter quantification. PLOS ONE [online]. 2022, **17**(10), e0275528. ISSN 1932-6203. Dostupné z: doi:10.1371/journal.pone.0275528

[19] BLENDER FOUNDATION. Blender 4.1 Manual [online]. [vid. 2024-04-07]. Dostupné z: https://docs.blender.org/manual/en/latest/index.html

[20] ECK, David J. Introduction to Computer Graphics [online]. Version 1.3. 2021. Dostupné z: http://sadil.ws/bitstream/handle/123456789/472/graphicsbook-linked.pdf?sequence=1&isAllowed=y

[21] HUGHES, John F. Computer graphics: principles and practice. Third edition. Upper Saddle River, New Jersey: Addison-Wesley, 2014. ISBN 978-0-321-39952-6.

[22] HORMANN, Kai a Marco TARINI. A quadrilateral rendering primitive. In: GH04: Graphics Hardware 2004: Proceedings of the ACM SIGGRAPH/EUROGRAPHICS conference on Graphics hardware [online]. Grenoble France: ACM, 2004, s. 7–14 [vid. 2024-04-23]. ISBN 978-3-905673-15-9. Dostupné z: doi:10.1145/1058129.1058131

[23] OHARA, Masakazu, Juno KIM a Kowa KOIDA. The Role of Specular Reflections and Illumination in the Perception of Thickness in Solid Transparent Objects. Frontiers in Psychology [online]. 2022, **13** [vid. 2024-04-24]. ISSN 1664-1078. Dostupné z: doi:10.3389/fpsyg.2022.766056

[24] PHONG, Bui Tuong. Illumination for Computer Generated Pictures. 1975, **18**(6).

[25] WOLFF, Lawrence B., Shree K. NAYAR a Michael OREN. Improved Diffuse Reflection Models for Computer Vision. International Journal of Computer Vision [online]. 1998, **30**(1), 55–71. ISSN 1573-1405. Dostupné z: doi:10.1023/A:1008017513536

[26] ASTUTI, Ika Asti, Ibnu Hadi PURWANTO, Tonny HIDAYAT, Dhimas Adi SATRIA, HARYOKO a Rizky PURNAMA. Comparison of Time, Size and Quality of 3D Object Rendering Using Render Engine Eevee and Cycles in Blender. In: 2022 5th International Conference of Computer and Informatics Engineering (IC2IE): 2022 5th International Conference of Computer and Informatics Engineering (IC2IE) [online]. 2022, s. 54–59 [vid. 2024-04-28]. Dostupné z: doi:10.1109/IC2IE56416.2022.9970186

[27] Introduction EEVEE - Blender 4.1 Manual [online]. [vid. 2024-04-28]. Dostupné z: https://docs.blender.org/manual/en/latest/render/eevee/introduction.html

[28] Introduction Cycles - Blender 4.1 Manual [online]. [vid. 2024-04-28]. Dostupné z: https://docs.blender.org/manual/en/latest/render/cycles/introduction.html

[29] Fibers_learning_set/script_fibers11.py at Dev · PM-GH-repo/Fibers_learning_set. GitHub [online]. [vid. 2024-05-06]. Dostupné z: https://github.com/PM-GH-repo/Fibers_learning_set/blob/Dev/script_fibers11.py

[30] SCHROEDER, William J., Ken MARTIN, William E. LORENSEN, Lisa Sobierajski AVILA, Kenneth W. MARTIN a Bill LORENSEN. The visualization toolkit: an object-oriented approach to 3D graphics; 4. ed. Clifton Park, NY: Kitware, Inc, 2006. ISBN 978-1-930934-19-1.

[31] RAMACHANDRAN, Prabhu a Gael VAROQUAUX. Mayavi: 3D Visualization of Scientific Data. Computing in Science & Engineering [online]. 2011, **13**(2), 40–51. ISSN 1558-366X. Dostupné z: doi:10.1109/MCSE.2011.35

[32] WOO, Mason, Mason WOO, a OPENGL ARCHITECTURE REVIEW BOARD, ed. OpenGL programming guide: the official guide to learning OpenGL, version 1.2. 3rd ed. Reading, MA: Addison-Wesley, 1999. ISBN 978-0-201-60458-0.

[33] BLENDER FOUNDATION. Blender Python API. Blender Python API Documentation [online]. [vid. 2024-04-07]. Dostupné z: https://docs.blender.org/api/current/?utm_medium=www-footer

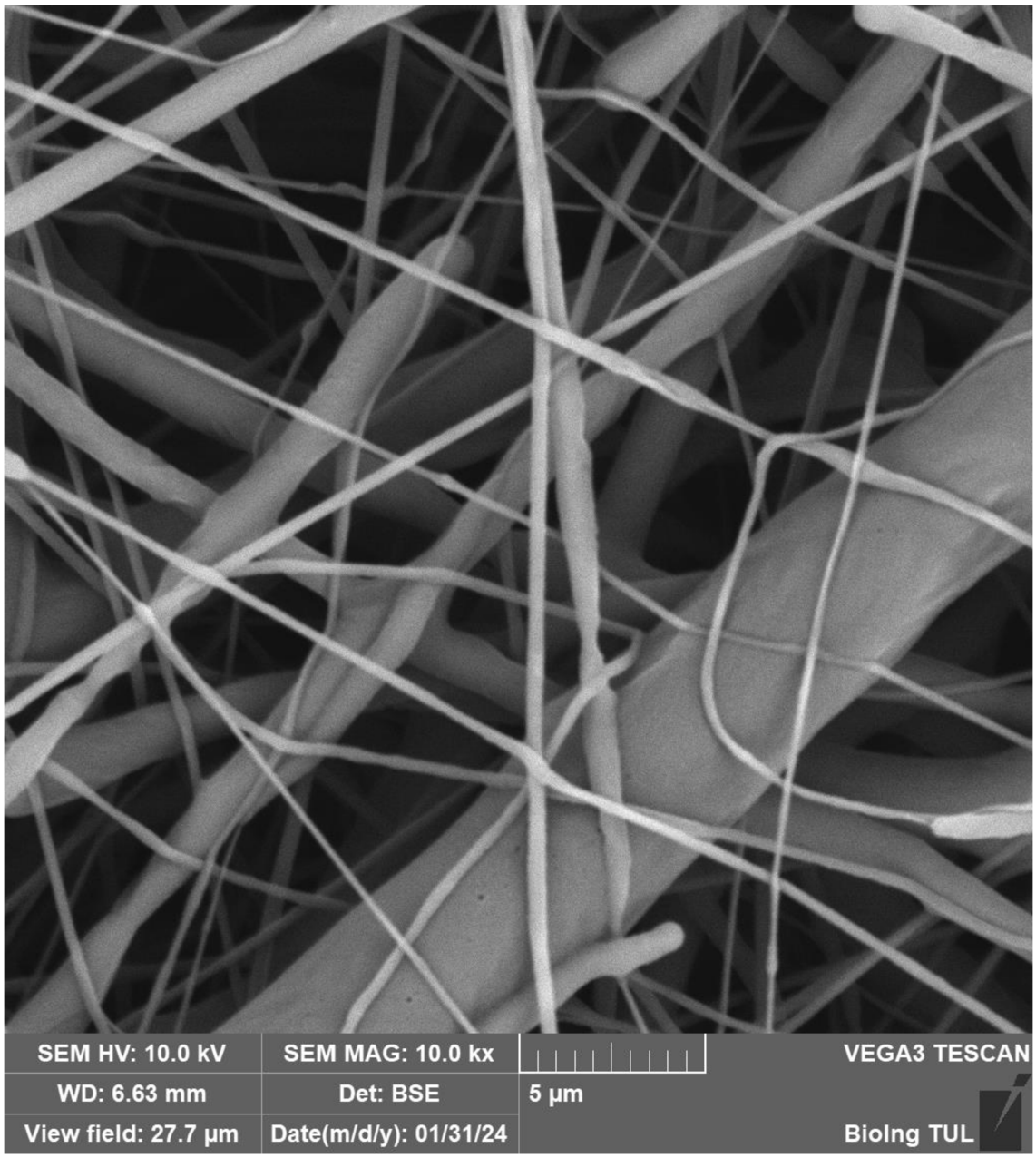
[34] RONNEBERGER, Olaf, Philipp FISCHER a Thomas BROX. U-Net: Convolutional Networks for Biomedical Image Segmentation [online]. B.m.: arXiv. 18. květen 2015 [vid. 2024-05-04]. Dostupné z: http://arxiv.org/abs/1505.04597. arXiv:1505.04597 [cs]

Přílohy

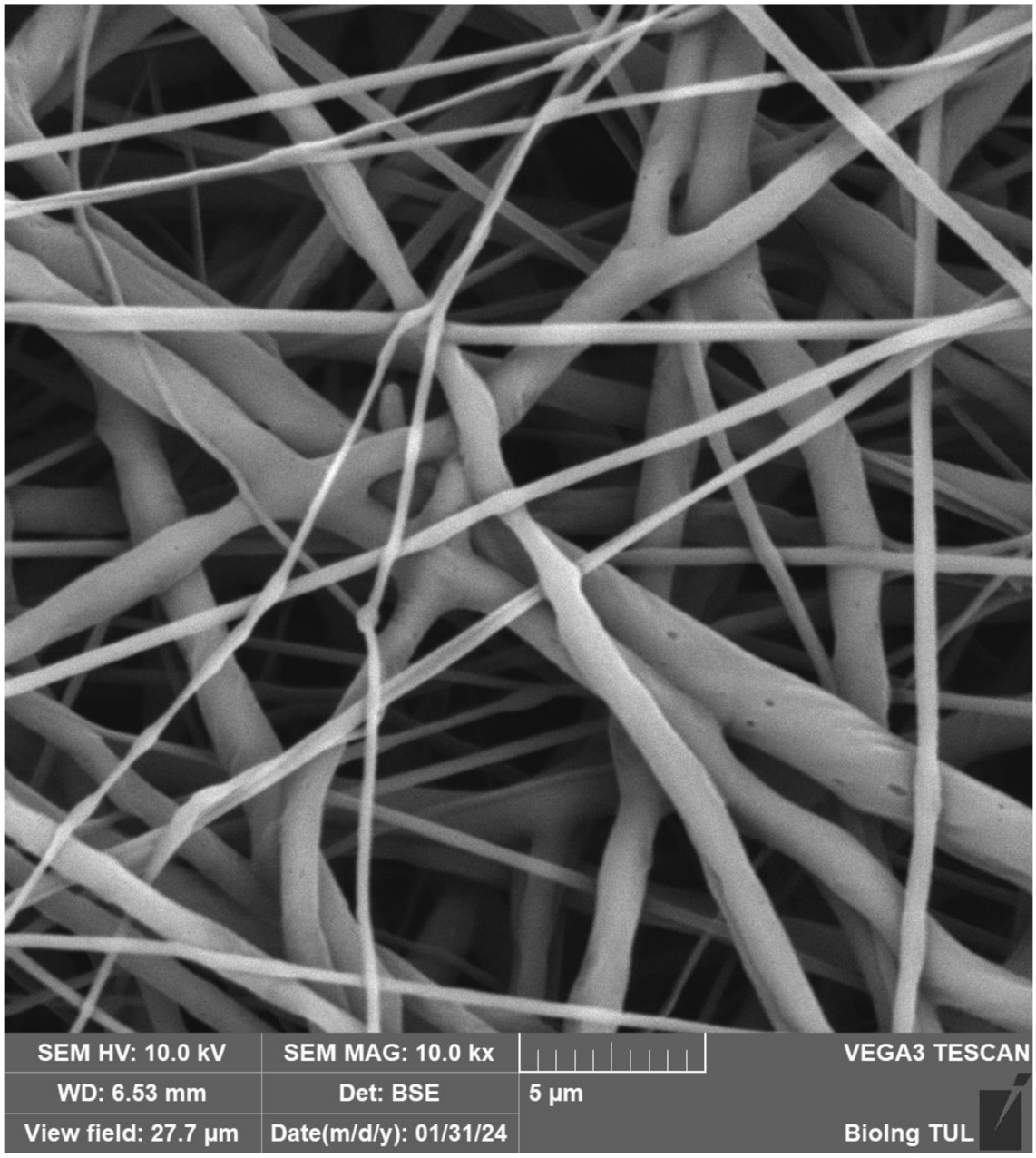
A Reálné snímky ze SEM (vzorky)

Na těchto snímcích bylo testováno, jak kvalitně určí síť průměr vlákna.

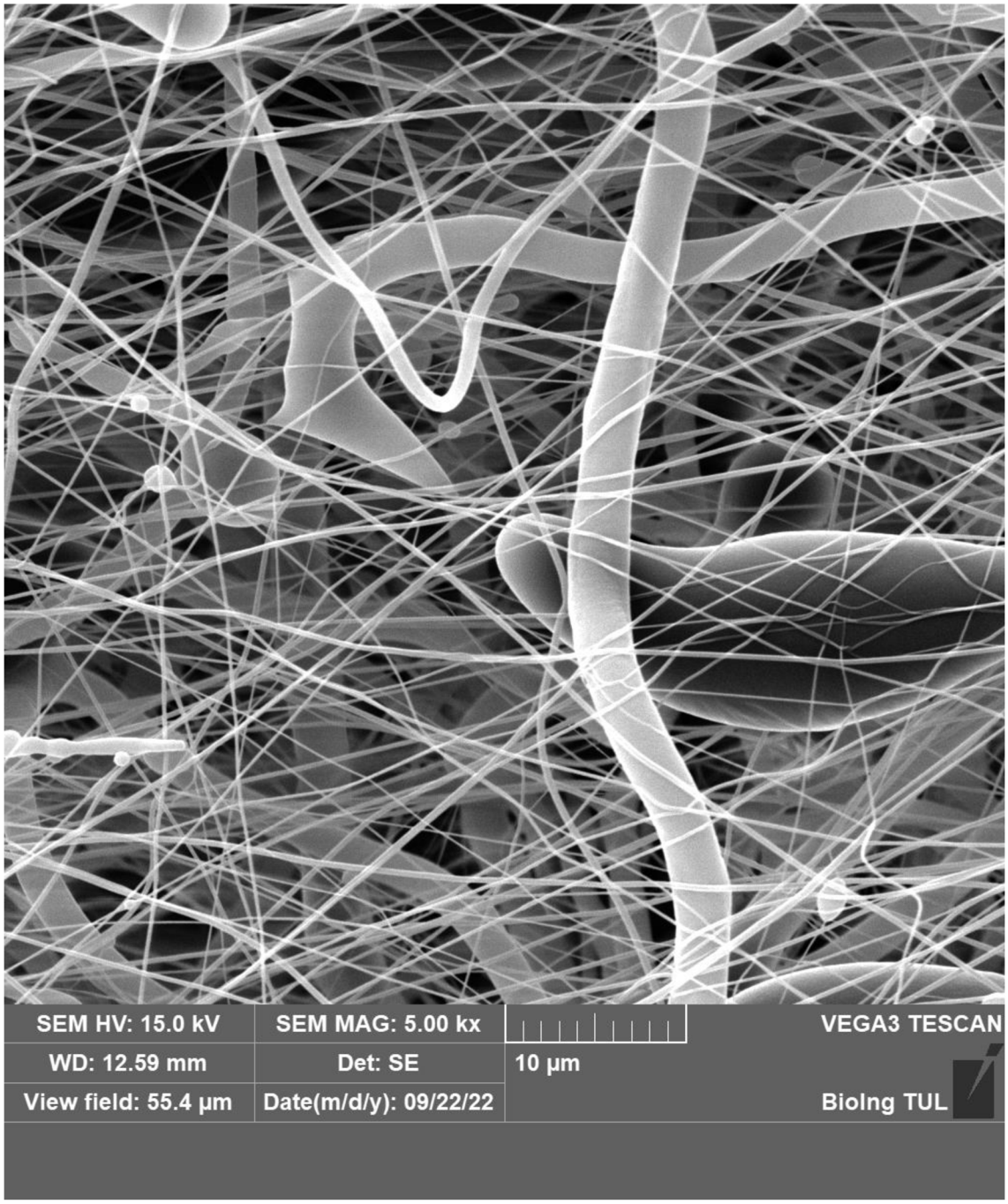
A.1 Vzorek ES_Pavla_PCL-dmf+AuNP_10kx_01xxBSE_true



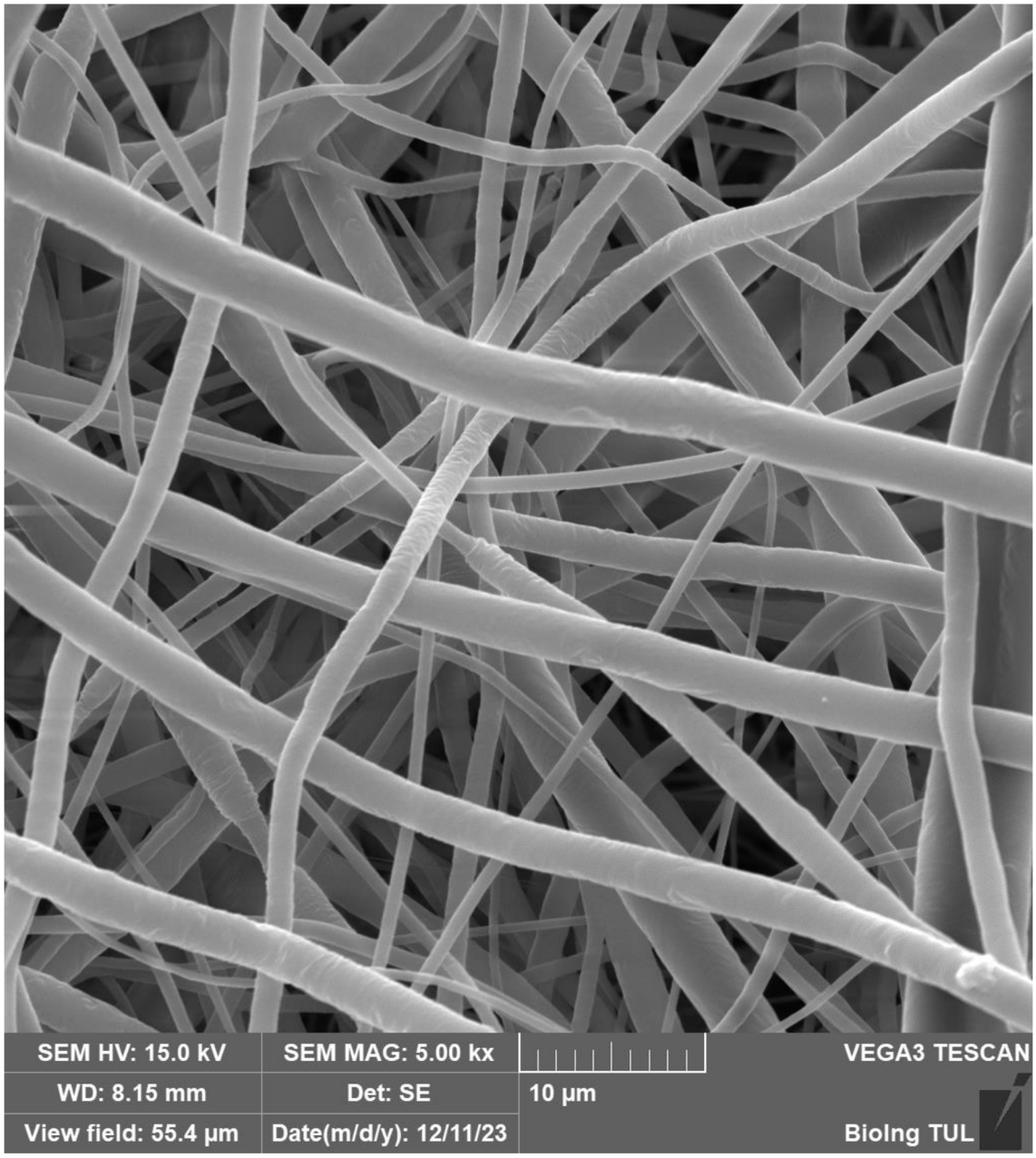
A.2 Vzorek ES_Pavla_PCL-etoh+AuNP_10kx_01xxBSE_true



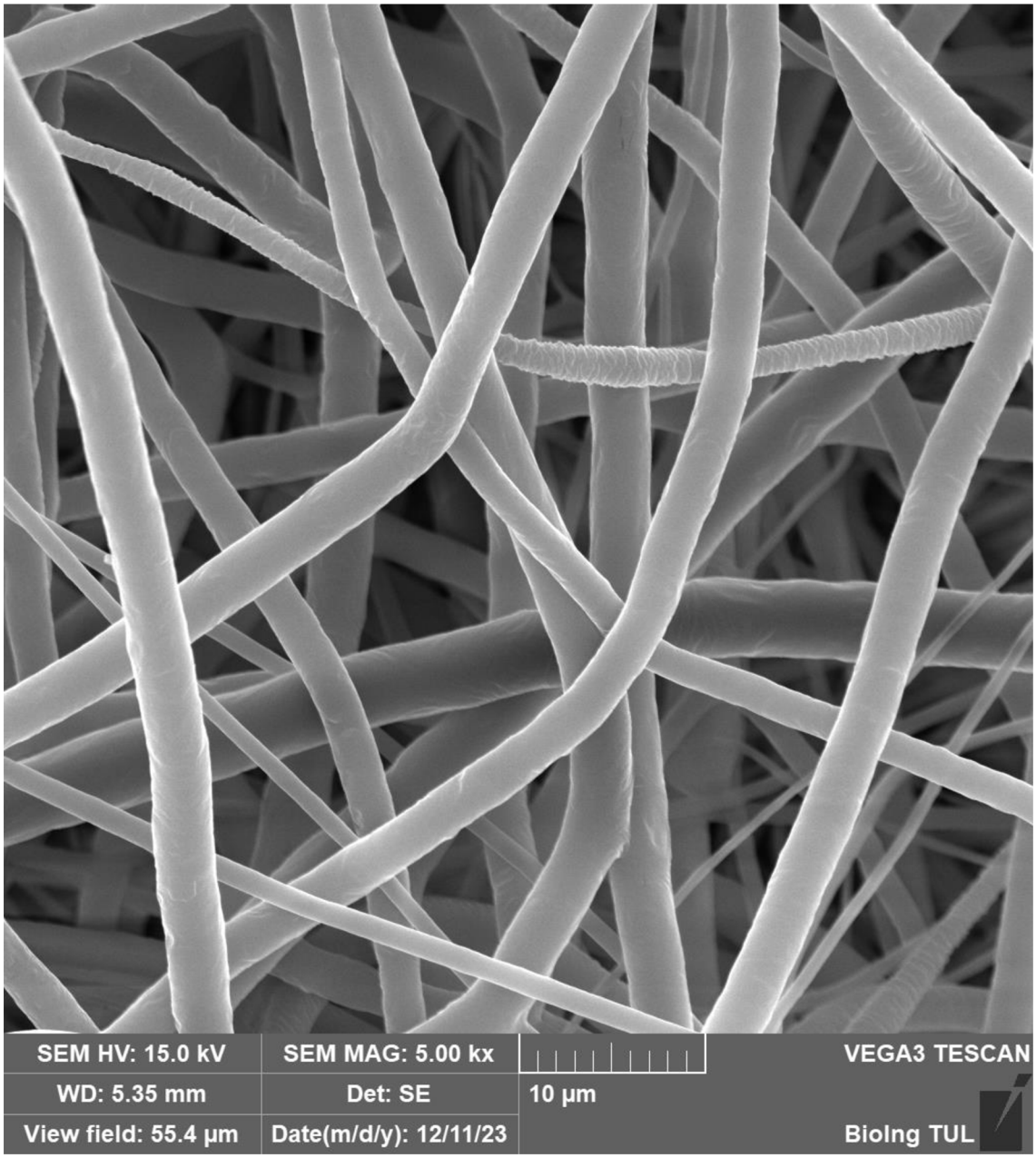
A.3 Vzorek nanospider_22,09,22_15mm_min_5kx_1



A.4 Vzorek PCL80-5kx



A.5 Vzorek PCL80-5kx2



B Příklad snímku sady B

Zbylé snímky sady (včetně anotací) jsou umístěn v souboru přílohy k diplomové práci.