

GYMNAZIUM A STŘEDNÍ ODBORNÁ ŠKOLA

ul. Mládežníků 1115, Rokycany



**VIZUALIZACE PROSTOROVĚ-FÁZOVÉHO
DIAGRAMU PLANÁRNÍHO OPTICKÉHO SYSTÉMU**

MATURITNÍ PRÁCE

Bronislav Růžička

Autor práce

Mgr. Helena Čížková

Vedoucí práce

Rokycany 2023

Prohlášení

Prohlašuji, že jsem maturitní práci s názvem „Vizualizace prostorovo-fázového diagramu planárního optického systému“ vypracoval samostatně pod vedením Mgr. Heleny Čížkové a uvedl v seznamu literatury všechny použité literární a odborné zdroje. Dále prohlašuji, že citace použitých pramenů je úplná a že jsem v práci neporušil autorská práva (ve smyslu zákona č. 121/2000 Sb. O právu autorském, o právech souvisejících s právem autorským).

V Rokycanech dne 28. února 2023



Bronislav Růžička

Poděkování

Děkuji Mgr. Heleně Čížkové za veškerou pomoc při zpracování této maturitní práce.
Rád bych poděkoval i svému konzultantovi Ing. Petru Lobazovi, Ph.D., za jeho odborné
rady z oblasti optiky a její simulace.

Obsah

1	Úvod	7
2	Materiál a metody	9
2.1	Vymezení problému	9
2.2	Analýza systému	13
2.2.1	Vztažná soustava	13
2.2.2	Znázornění chodu paprsků	13
2.2.3	Definice prostorovo-fázového diagramu	14
2.2.4	Vlastnosti prostorovo-fázového diagramu	16
2.2.5	Étendue	21
2.2.6	Graf rozložení svítivosti	25
2.3	Implementační prostředky	27
2.3.1	Programovací jazyk	27
2.3.2	Vývojové prostředí	27
2.4	Architektura a funkce aplikace	28
2.4.1	Typické použití aplikace	28
2.4.2	Zadání scény	28
2.4.3	Simulace paprsků	30
2.4.4	Vizualizace scény	31
2.4.5	Detekce paprsků	33
2.4.6	Vizualizace prostorovo-fázového diagramu	34
2.4.7	Vizualizace grafu rozložení svítivosti	35
2.5	Druhy členů scény	36
2.5.1	Světelné zdroje	36
2.5.2	Typy optických prvků	38
2.5.3	Geometrie optických prvků	39
3	Výsledky	41
3.1	Světlovody	41
3.1.1	Krátký světlovod	41
3.1.2	Dlouhý světlovod	43

3.2 Parabolické reflektory	44
3.2.1 Parabolický reflektor s ohniskem ve středu vstupního otvoru . . .	44
3.2.2 Parabolický reflektor s ohniskem za vstupním otvorem	47
3.2.3 Parabolický reflektor s ohniskem před vstupním otvorem	48
4 Diskuze	50
4.1 Celková užitečnost	50
4.2 Nedostatky návrhu aplikace	50
4.3 Nedostatky implementace aplikace	51
5 Závěr	53

Abstrakt

Maturitní práce *Vizualizace prostorovo-fázového diagramu planárního optického systému* byla vytvořena se záměrem vyhodnotit potenciální užitečnost takového nástroje pro oblast návrhu optiky. Zejména pak pro odvětví nezobrazovací optiky, jejíž návrh je i dnes značně složitý. Hlavní částí práce bylo pomocí vhodných prostředků sestavit jednoduchou počítačovou aplikaci schopnou pro zadaný optický systém zobrazit jednak zmíněný prostorovo-fázový diagram, ale i související analýzy, tj. výpočet étendue a vykreslení grafu rozložení svítivosti. V této práci se podařilo prokázat, že tento koncept není možné přehlížet, a bylo by vhodné se tímto problémem zabývat i nadále.

1 Úvod

Optika a *optický systém* jsou pojmy, pod kterými si nejčastěji vybavíme objektiv fotoaparátu, dalekohled nebo brýle. Ovšem realita tohoto odvětví fyziky je poněkud složitější, celé se totiž rozděluje několik různých částí.

Velká část středoškolských znalostí, co se světelných soustav týče, se řadí do kategorie *zobrazovací optiky*. Spadají do ní i všechny výše uvedené předměty a obecně se specializuje na vytváření obrazu nějaké předlohy.

Jistý protipól tvoří *nezobrazovací optika*, ta se zabývá spíše distribucí světla od zdroje určitým zadaným způsobem. Její systémy jsou podstatně rozšířenější, ale zato zpravidla mnohem méně nápadné. Počítáme mezi ně nejrůznější svítily, světlomety aut, reflektory v divadlech, osvětlení budov, ale i optická vlákna a solární panely nebo např. displeje. (Zde je dobré si uvědomit, že na displeji sice vidíme jakýsi obraz. Ten je ovšem tvořen pouze rozprostřením barevných bodů, pixelů. Optické prvky sice se používají k podsvícení a k distribuci paprsků z jednotlivých pixelů do prostoru, ale ne přímo pro vytvoření obrazu.)

Historicky se optické systémy navrhovaly zejména metodou *pokus-omyl*, dokud se dosahované výsledky nepřiblížily záměru (srov. první dalekohledy a mikroskopy na začátku 17. století). V této době neexistovaly v podstatě téměř žádné pomůcky ani vzorce, na základě kterých by se daly potřebné tvary čoček a zrcadel vypočítat.

Pro *zobrazovací optiku* nastala změna v průběhu 19. století. Díky novým objevům bylo umožněno matematicky modelovat různé optické soustavy a přejet tak slepému zkoušení různých konfigurací. (Hannavy, 2008; Wimmer, 2017)

Naopak *nezobrazovací optika* žádnou podobnou revoluci neprodělala a i dnes bývá proces navrhování takových systémů z větší části založen na náhodném zkoušení možností. To je způsobeno zejména její značnou složitostí. Jako nečekaně komplexní se prokazuje i tato základní úloha:

Navrhněte reflektor o maximálních rozměrech $20 \times 20 \times 20 \text{ mm}^3$ pro LED světelny zdroj $3 \times 3 \text{ mm}^2$, který vytvoří kužel světla o šířce 10° .

Nejenže neexistuje žádný přímý způsob, jak se jednoduše dobrat výsledku. Se současnými metodami ani nedokážeme rozhodnout, jestli je úloha vůbec řešitelná. (P. Lobaz,

osobní komunikace, 11. 2. 2023)

Obecně je při návrhu optické soustavy potřeba nejprve problém zjednodušit a vymyslet základní koncept řešení, které se posléze dále vylepšuje. Při práci se *zobrazovacím systémem* často používáme idealizované tenké čočky, dalším způsobem může být výpočet pouze na základě několika významných paprsků apod.

Obdobných pomůcek z oblasti *nezobrazovací optiky* není mnoho. A pokud existují, zpravidla nejsou dostupné v žádném komerčním softwaru pro návrh optických soustav, např. *TracePro* (lambdares.com/tracepro), *LightTools* nebo *Photopia* (ltioptics.com).
(P. Lobaz, osobní komunikace, 11. 2. 2023)

Jedním z nástrojů, který by mohl pomoci zjednodušit a urychlit proces návrhu nezobrazovacích systémů, je i vizualizace tzv. *prostorovo-fázového diagramu* (anglicky *phase-space diagram*). (Muschaweck et al., 2022) Cílem této práce je zjistit, zda-li počítačový program, který umožňuje interaktivně zobrazit právě takovýto diagram, dokáže přispět k lepšímu pochopení navrhované soustavy a případně k jejímu vylepšení.

2 Materiál a metody

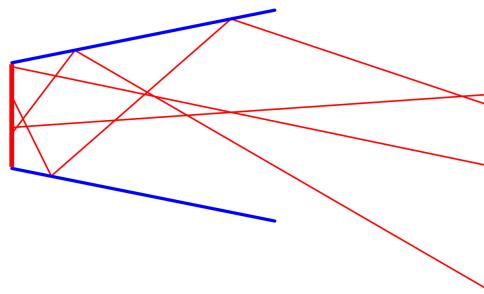
2.1 Vymezení problému

Smyslem této práce je pouze demonstrovat jistý koncept. Zaměříme se proto jen na úplné jádro problému, a pokud se výsledek prokáže jako užitečný, můžeme uvažovat o nějakém zobecnění. Připomeňme si, že cílem je jednak samozřejmě pochopení celkové funkce navrhovaného systému, ovšem neméně nás zajímá i charakteristika vystupujícího světla.

Příkladem, na kterém si ukážeme postupné zjednodušení, budiž tento jednoduchý světlovod, jenž bude mít tvar *pláště komolého jehlanu (případně hranolu)* s *odříznutými podstavami*. Namísto horní podstavy umístíme *plošný zdroj světla*. Vysílané paprsky se z něj šíří skrz jehlan, a jak přímo, tak s pomocí odrazů od stěn.

V prvním kroku se omezíme pouze na podélný řez této soustavy, jinými slovy, budeme se pohybovat pouze ve 2D. Zdrojem světla bude nyní jen úsečka a celý světlovod se zjednoduší na dvě úsečky reprezentující zrcadla. To je znázorněno na následujícím obrázku.

Pro zajímavost také uvedeme, že veškeré obrázky optických soustav i souvisejících diagramů byly vygenerovány programem, jenž tvoří hlavní část této práce.



Obrázek 1: Jednoduchý 2D světlovod tvořený dvěma zrcadly (modrá) s úsečkovým zdrojem světla a několika znázorněnými paprsky (červená)

V souvislosti s tímto zjednodušením je nutné podotknout, že výsledky simulace ve 2D se mohou v určitých směrech zásadně lišit od těch pro 3D. Jako jeden z rozdílů můžeme

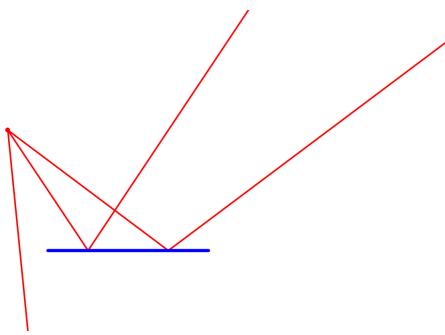
uvést fyzikální veličinu osvětlenost ($[E] = 1 \frac{\text{lm}}{\text{m}^2}$). Ve třech prostorových dimenzích se snižuje s druhou mocninou vzdálenosti (podle zákona převrácených čtverců, anglicky *inverse-square law*), zatímco její dvourozměrný protějšek se bude zmenšovat pouze lineárně.

Zároveň zdůrazněme, že nebudeme nahrazovat *plošný zdroj* světla *bodovým*. Tato metoda je velmi typická pro zobrazovací optiku, ale v oblasti nezobrazovací optiky se příliš nepoužívá. Hlavním důvodem je velikost světelých zdrojů, která bývá v navrhovaných systémech zpravidla nezanedbatelná vůči velikosti celé soustavy.

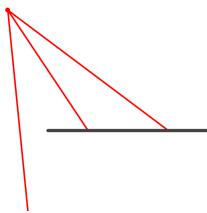
Dále se omezíme na studium makroskopických soustav (svítilna, světlomet). Budeme se zajímat především o situace, ve kterých se soustavou šíří velké množství paprsků (ideálně nekonečné). To je další rozdíl oproti zobrazovací optice, kde k pochopení funkce systému stačí znalost chodu maximálně nižší desítky paprsků. (Chaves, 2017) Zároveň v takto velkých systémech se v podstatě neprojevují (resp. projevují, ale zanedbatelně) *vlnové vlastnosti světla*, které proto ze simulace můžeme vypustit. Ve výsledku tedy budeme na světlo nahlížet pouze z pohledu *paprskové optiky*, což by nebylo možné např. při zkoumání optických vláken, které se svými rozměry více přibližují vlnové délce použitého světla.

Budeme předpokládat, že všechny paprsky mají stejný, *elementární*, optický výkon. Optický výkon celého světelného zdroje je pak jasně dán pouhým počtem paprsků, které vyzařuje.

Pro jednoduchost budeme v demonstračním programu pracovat pouze se dvěma typy interakcí mezi systémem a paprskem, *dokonalým zrcadlovým odrazem* a *dokonalou absorpcí*. Pro praktické využití by se hodilo zahrnout i *lom světla*, *rozptyl* (odraz od matného povrchu), *částečný odraz* aj. (Žára et al., 2005) Zanedbáme též závislost zmíněných jevů na polarizaci nebo vlnové délce (barvě) světla.

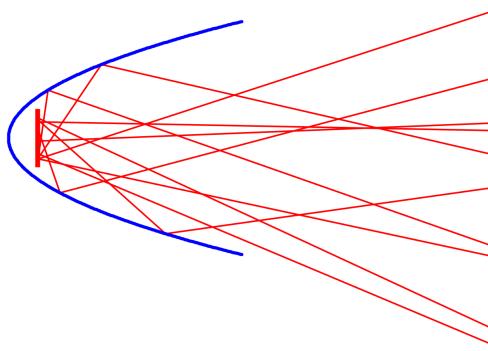


Obrázek 2: Příklad dokonalého odrazu na zrcadle (modrá) s bodovým zdrojem světla a několika znázorněnými paprsky (červená)



Obrázek 3: Příklad dokonalé absorpcie na cloně (šedá) s bodovým zdrojem světla a několika znázorněnými paprsky (červená)

Reálné optické prvky bývají ve většině případů tvarované na základě nejrůznějších matematických popisů (např. $y = k \cdot x^2$). Ovšem jelikož výpočty s mnoha typy křivek a funkcí nebývají vždy snadné, budeme připouštět pouze approximaci těchto tvarů pomocí lomených čar (série mnoha krátkých úseček). Při budoucím rozvoji bylo vhodné podporovat alespoň některé hojně používané tvary, např. parabola, *Bézierovy křivky*.
(Žára et al., 2005)



Obrázek 4: Aproximace parabolického zrcadla (modrá) s úsečkovým zdrojem světla a několika znázorněnými paprsky (červená)

V nezobrazovací systémech zpravidla není zřejmé, jak přesně bude který paprsek postupovat optickou soustavou. To je podstatný rozdíl od zobrazovací optiky kde je to velmi jednoduše vyvoditelné. Jako příklad si vezměme teleskop se dvěma čočkami. U něj je na první pohled jasné, že paprsek světla nejdříve projde přes přední, a pak zadní plochu první čočky, a poté totéž pro čočku druhou. Zejména díky této znalosti je analýza i návrh zobrazovacích soustav podstatně jednodušší, než je tomu u systémů nezobrazovacích.

2.2 Analýza systému

2.2.1 Vztažná soustava

Nejprve je nutné si zvolit vztažnou soustavu pro analýzu systému. Použijeme *kartézskou soustavu souřadnic* s počátkem O a osami x a y (soustava Oxy). Optický systém do ní umístíme tak, aby se většina paprsků šířila ve směru $+x$. Poté lze (též) libovolný paprsek p jednoznačně popsat následující rovnicí:

$$p : y = k \cdot x + q$$

Přičemž x a y jsou souřadnice bodů, jimiž paprsek prochází, q je y -souřadnice průsečíku paprsku se zvolenou osou y a k je směrnice paprsku. Ta se dá definovat i v závislosti na úhlu α mezi paprskem a osou x :

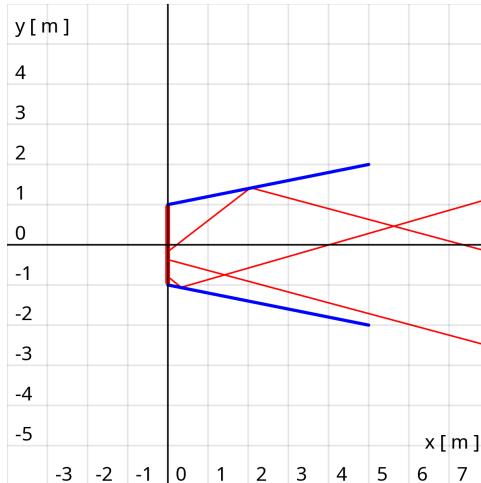
$$k = \tan \alpha$$

Problémy s nejednoznačností nastávají, když se některé paprsky šíří ve směru $-x$. Zároveň paprsky rovnoběžné s osou y nelze pomocí výše uvedené rovnice definovat. Budeme se proto zabývat pouze systémy, pro než není analýza ve směru osy y zásadně důležitá, popř. můžeme celý optický systém otočit o 90° .

2.2.2 Znázornění chodu paprsků

První částí analýzy světelného systému je neodmyslitelně prozkoumání samotné geometrie optických prvků a chodu jednotlivých paprsků. Toho docílíme pomocí zobrazení jednoduché 2D vizualizace scény. Použijeme již zvolenou soustavu Oxy a pouze doplníme, že na obou osách budou (libovolné) jednotky délky. V našem případě zvolíme metry (m).

Jako příklad poslouží jednoduchý světlovod již jednou popsaný na začátku kapitoly 2.1 Vymezení problému:



Obrázek 5: Jednoduchý 2D světlovod tvořený dvěma zrcadly (modrá) s úsečkovým zdrojem světla a několika znázorněnými paprsky (červená), zakreslený v systému Oxy (černá a šedá)

2.2.3 Definice prostorovo-fázového diagramu

Prostorovo-fázový diagram (pf-diagram) (Muschaweck et al., 2022) zkoumá chování paprsků v blízkém okolí osy y . Pf-diagram je kartézský, tj. má vodorovnou a svislou osu. Každý jeho bod jednoznačně určuje nějaký paprsek.

Svislá souřadnice těchto bodů je rovna hodnotě q z předchozí definice, neboli souřadnici průsečíku paprsku s osou y . Z toho vyplývá, že jednotky (a pro uživatelskou přivětivost často i měřítko) svislé osy pf-diagramu budou shodné jako u osy y systému Oxy .

Naproti tomu, vodorovná souřadnice vyjadřuje sklon paprsku, ať už ve formě úhlu α , směrnice k , nebo sinu úhlu $\sin \alpha$ apod.

V této části práce budeme využívat diagramy, které na vodorovné ose zobrazují směrnici k . Jinými slovy, pro n -tý paprsek p_n , definovaný jako:

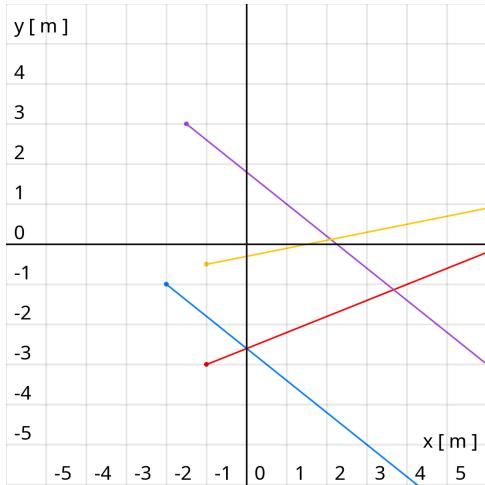
$$p_n : y = k_n \cdot x + q_n$$

zakreslíme do pf-diagramu bod X_n :

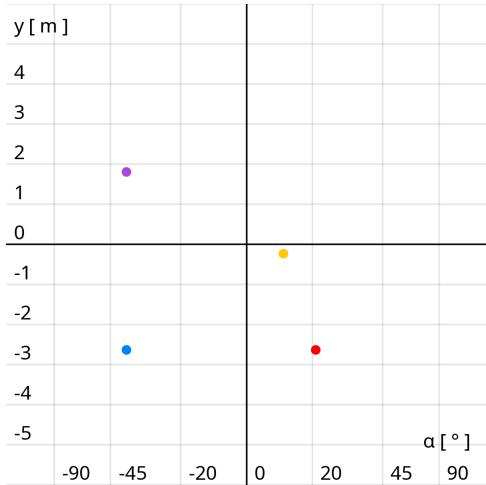
$$X_n = [k_n, q_n]$$

V dalším textu budeme proto vodorovnou osu pf-diagramu nazývat k a svislou osu q .

Jednoduchý příklad pf-diagramu je znázorněn na následujících obrázcích:



Obrázek 6: Scéna s několika barevně označenými paprsky



Obrázek 7: Vizualizace paprsků z předchozího obrázku v pf-diagramu

Všimněme si, že modrý paprsek je rovnoběžný s paprskem fialovým. To má za následek shodnost (vodorovných) k -souřadnic bodů, jež tyto paprsky reprezentují v pf-diagramu. Podobně, průsečík červeného paprsku s osou y je totožný s průsečíkem modrého paprsku s osou y . To způsobuje, že odpovídající body mají stejnou q -souřadnici.

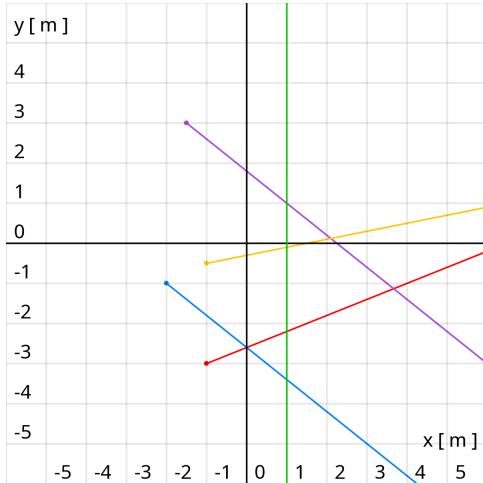
Pro lepší pochopení optických systémů se hodí možnost odpoutat pf-diagram od osy y a vypočítávat jej pro libovolnou s ní rovnoběžnou přímku. Takovouto *detekční přímku* lze definovat:

$$d : \quad x = c$$

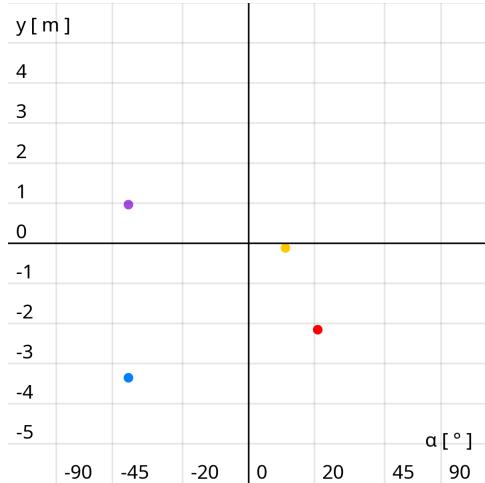
kde c je konstanta, o kterou je přímka posunutá od osy y . Bod odpovídajícího pf-diagramu pak má souřadnice:

$$X_n = [k_n, p_n(c)]$$

Příkladem bude tatáž scéna jako výše, ovšem s detekční přímkou pro $x = 1$ m:



Obrázek 8: Scéna s několika barevně označenými paprsky a detekční přímkou (zelenou) pro $x = 1$ m



Obrázek 9: Vizualizace paprsků z předchozího obrázku v pf-diagramu pro detekční přímku $x = 1$ m

V diagramu samozřejmě zobrazujeme pouze ty sekce paprsků, které opravdu protínají detekční přímku.

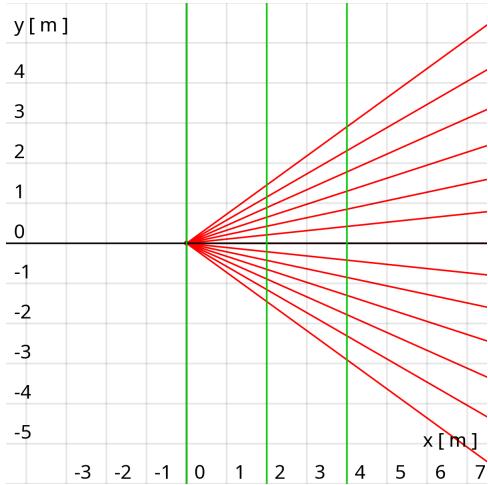
Je také vhodné zmínit, že pro 2D simulace je pf-diagram, jak již víme, taktéž dvourozměrný. Ovšem pro 3D systémy potřebujeme k zobrazení pf-diagramu dimenze celkem 4. Dvě prostorové veličiny, pro přesné určení průsečíku s detekční rovinou, a dvě úhlové pro jednoznačné určení směru paprsku. To výrazně komplikuje vizualizaci, což je i jeden z dalších důvodů, proč se v této práci omezujeme jen na 2D systémy. Lze předpokládat, že pokud bude rovinná analýza neužitečná, bude těžko použitelný i rozbor v prostoru.

2.2.4 Vlastnosti prostorovo-fázového diagramu

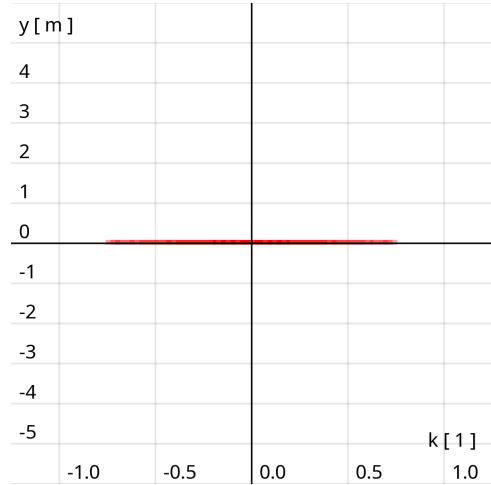
Pokud detekční přímka prochází přímo *bodovým zdrojem* (resp. *vějířovým*, uvažujeme-li pouze paprsky šířící se zleva doprava), zobrazí se na pf-diagramu vodorovná úsečka. Všechny paprsky jsou totiž koncentrovány do jednoho bodu, průsečíku, zatímco jejich směrnice se podstatně liší. To je patrné i na následujících dvou obrázcích.

Poukažme na to, že pro nakreslení přesného pf-diagramu je nutné počítat s desítkami tisíc až miliony paprsků. Ovšem v obrázcích s optickou soustavou je zobrazení takového množství paprsků naprosto nepřehledné, proto do nich vždy zakreslujeme pouze malý

výběr z celkového počtu.

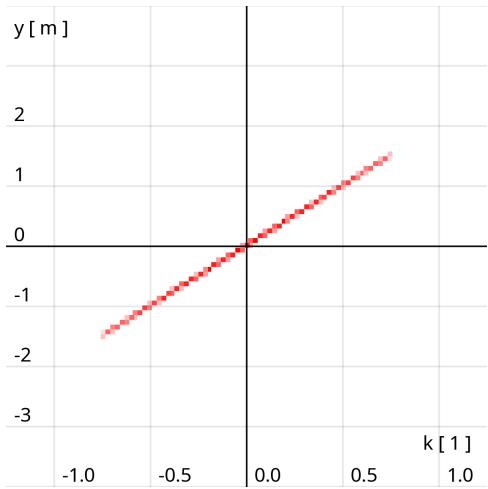


Obrázek 10: Vějířový zdroj světla s několika znázorněnými paprsky (červená) a tři vyzačené detekční přímky, pro $x = 0$ m, $x = 2$ m a $x = 4$ m (zelená)

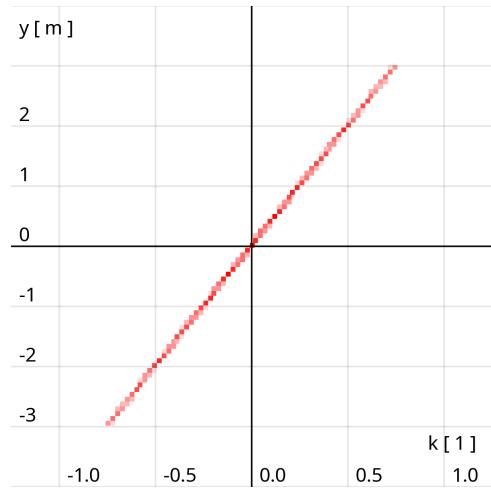


Obrázek 11: Vizualizace pf-diagramu pro levou detekční přímku $x = 0$ m v předchozím obrázku, procházející zdrojem světla

Pokud budeme posouvat detekční přímku dále, nastane jistá deformace obrazce v pf-diagramu, přesněji, tvar se zkosi (to ovšem platí pouze pro diagram typu *směrnice-průsečík*). Důvodem této transformace je, že čím dál od osy y pf-diagramu se promítne, tím větší má sklon a tím rychleji se posouvá dále od osy x .

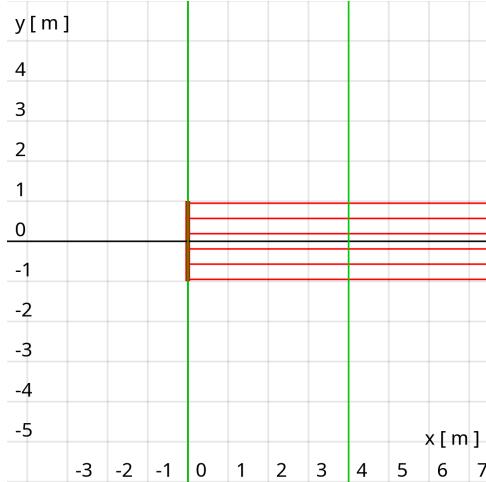


Obrázek 12: Vizualizace pf-diagramu pro prostřední detekční přímku $x = 2$ m v předchozím obrázku, znázorňuje zkosení obrazce

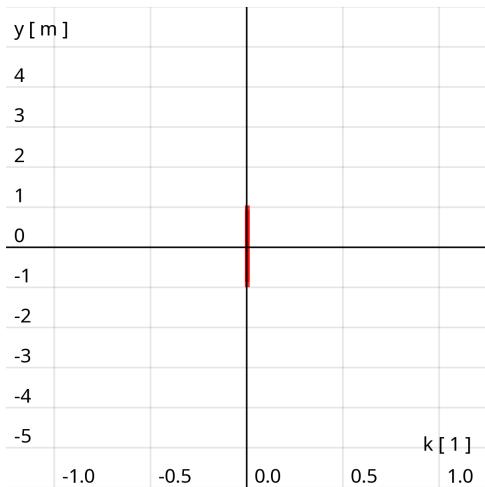


Obrázek 13: Vizualizace pf-diagramu pro pravou detekční přímku $x = 4$ m v předchozím obrázku, znázorňuje zkosení obrazce

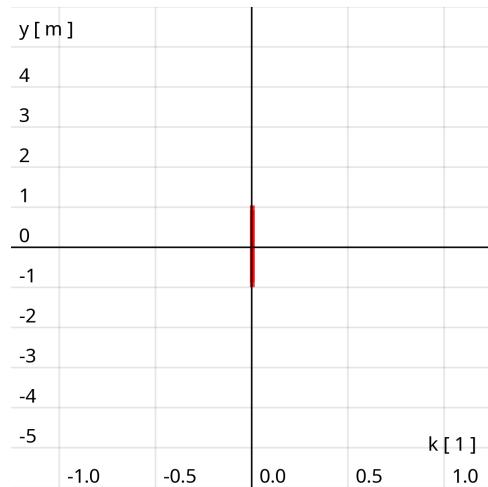
Oproti tomu zdroj, jenž vyzařuje rovnoběžný svazek paprsků, se zobrazí jako svislá úsečka. Paprsky mají stejný sklon, proto se promítou pod sebe. To ilustrují následující obrázky:



Obrázek 14: Zdroj rozvoběžného svazku paprsků, z nichž některé jsou znázorněny (červená), a dvě vyznačené detekční přímky, pro $x = 0 \text{ m}$ a $x = 4 \text{ m}$ (zelená)



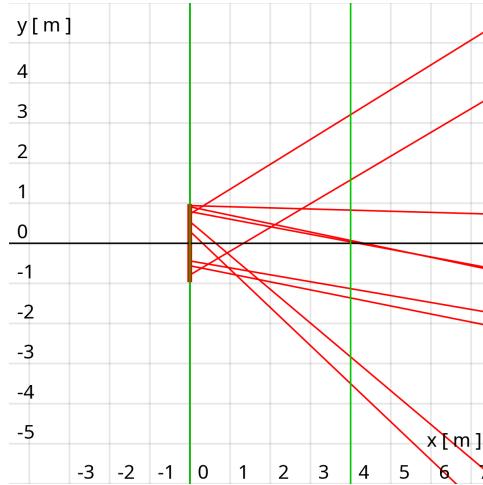
Obrázek 15: Vizualizace pf-diagramu pro levou detekční přímku $x = 0 \text{ m}$ v předchozím obrázku, procházející zdrojem světla



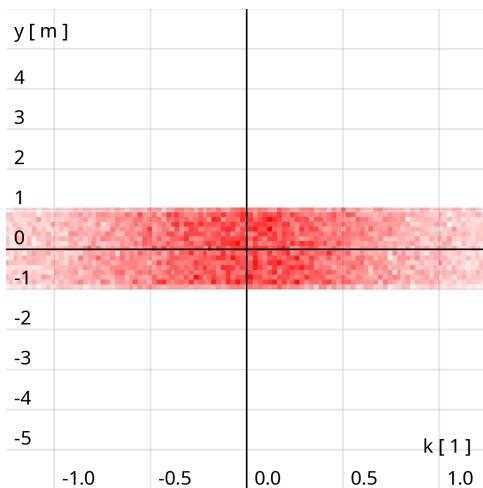
Obrázek 16: Vizualizace pf-diagramu pro pravou detekční přímku $x = 4 \text{ m}$ v předchozím obrázku, bez změny oproti diagramu procházejícímu zdrojem světla

Specificky pro tento případ, kdy mají paprsky nulový sklon, zůstává pf-diagram neméně bez ohledu na vzdálenost detekční přímky od zdroje. Při návrhu systému s nízkým rozptylem se snažíme co nejvíce přiblížit právě takovému rozložení.

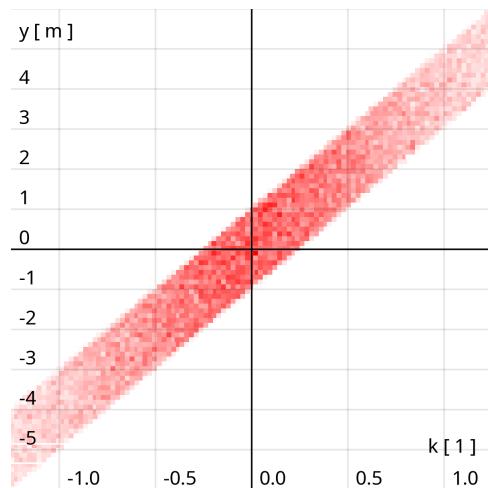
Pokud zkombinujeme obě zmíněné zákonitosti, dokážeme odvodit pf-diagramy i pro složitější zdroje světla, např. pro *plošný zdroj*, resp. *úsečkový* ve 2D. Základní obrazec, který tento záříč v pf-diagramu vytváří, je obdélník a při posouvání detekční přímky opět dochází ke zkosení. To lze pozorovat na následujících obrázcích.



Obrázek 17: Úsečkový zdroj světla s několika znázorněnými paprsky (červená) a dvě vyznačené detekční přímky, pro $x = 0$ m a $x = 4$ m (zelená)



Obrázek 18: Vizualizace pf-diagramu pro levou detekční přímku $x = 0$ m v předchozím obrázku, procházející zdrojem světla



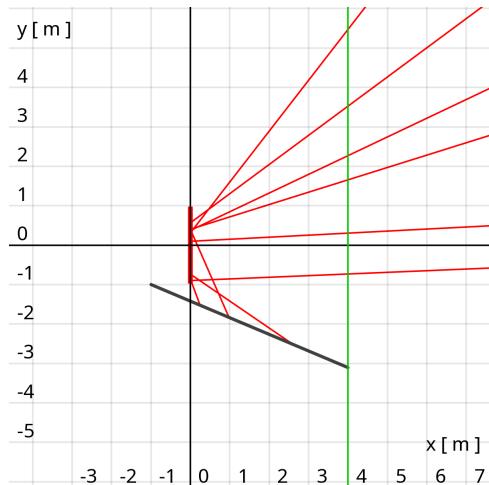
Obrázek 19: Vizualizace pf-diagramu pro pravou detekční přímku $x = 4$ m v předchozím obrázku, znázorňuje zkosení obrazce

Pro levou detekční přímku a levý pf-diagram platí, že bodem na detekční přímce (odopavidá vodorovné čáře v pf-diagramu) buďto neprochází žádný paprsek (např. $y = 2$ m), nebo jím naopak prochází paprsky všemi směry (např. $y = 0$ m).

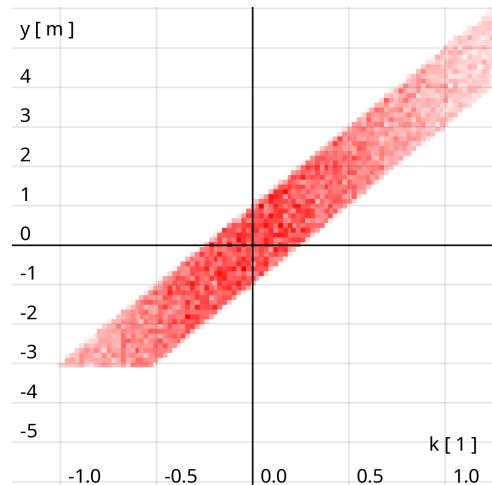
Pro pravou detekční přímku a pravý pf-diagram platí, že každým bodem na detekční přímce prochází paprsky v omezeném sklonovém (úhlovém) rozsahu. Např. bodem $y = 0 \text{ m}$ prochází pouze paprsky se směrnicemi v intervalu $k \in (-0.25, 0.25)$ a obdobně bodem $y = 1 \text{ m}$ pouze paprsky se směrnicemi $k \in (0.0, 0.5)$. Z diagramu lze taktéž vyčíst, že paprsky se sklonem $\alpha = 45^\circ$, čemuž odpovídá směrnice $k = 1$, protínají detekční přímku jen v rozmezí od $y_{min} = 3 \text{ m}$ do $y_{max} = 5 \text{ m}$.

Dále je nutno též dodat, že pod pojmy plošný resp. úsečkový zdroj obecně rozumíme tzv. *lambertovské zářiče* (taktéž *kosinové zářiče*) odpovídajícího tvaru. (*Fotometrie a radiometrie*, 2006) Tj. zářič, u něhož je počet vyzářených paprsků určitým směrem přímo úměrný jeho zdánlivé ploše při pohledu z daného směru. Díky použití těchto zdrojů můžeme v pf-diagramu pozorovat nerovnoměrné rozprostření paprsků. To je naznačeno různou sytostí barvy jednotlivých bodů. Platí čím je barva sytější, tím více paprsků je daným pixelem reprezentováno.

Zatím jsme rozebrali systémy skládající se pouze ze zdrojů světla. Nyní se podíváme, jak pf-diagram reaguje na optické prvky. Nejprve se zaměříme na clonu. To je docela jednoduché, část obrazce v diagramu je jednoduše uříznuta, bez nahradby (srov. s předchozími diagramy):

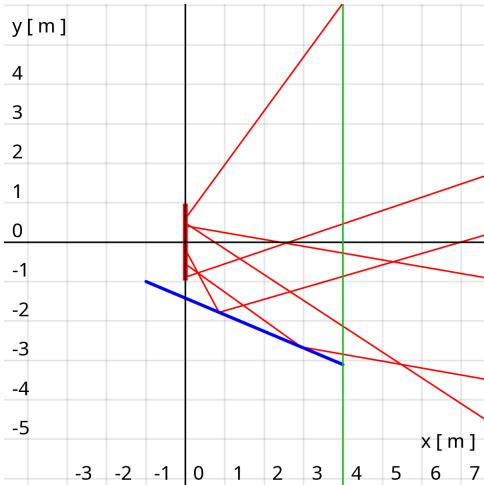


Obrázek 20: Úsečkový zdroj světla s několika znázorněnými paprsky (červená), clona (šedá) a vyznačená detekční přímka $x = 4 \text{ m}$ (zelená)

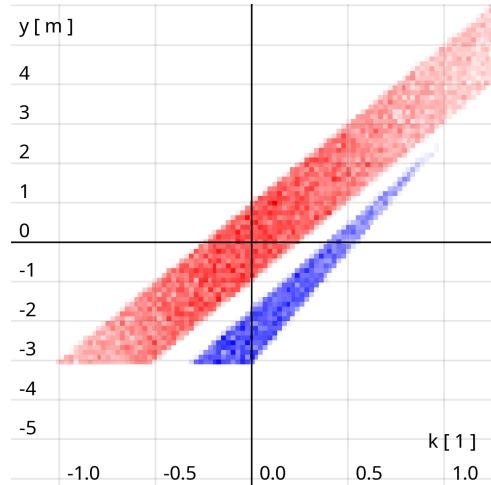


Obrázek 21: Vizualizace pf-diagramu scény na předchozím obrázku

Pro zrcadla je situace o něco složitější, ty totiž, obdobně jako clony, část obrazce odříznou, ale na rozdíl od nich tento kus zdeformují a přesunou jej na jiné místo v diagramu. Příklad můžete vidět na následujících obrázcích:



Obrázek 22: Úsečkový zdroj světla s několika znázorněnými paprsky (červená), zrcadlo (modrá) a vyznačená detekční přímka $x = 4$ m (zelená)



Obrázek 23: Vizualizace pf-diagramu scény na předchozím obrázku, obsahuje neodražené paprsky (červená) a odražené paprsky (modrá)

2.2.5 Étendue

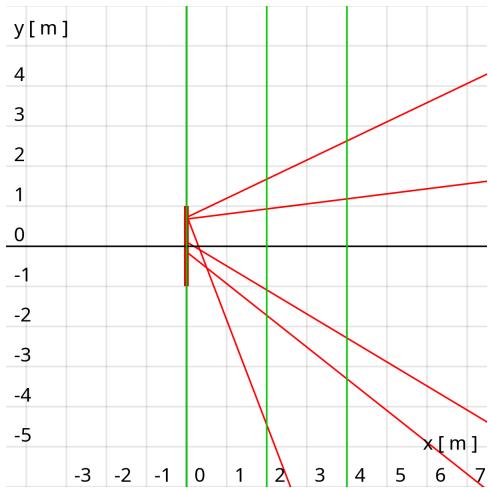
Doted'jsme se zabývali pf-diagramem typu *směrnice-průsečík*, a to zejména pro jednoduchost pochopení základních konceptů a vlastností. Ovšem zmiňovali jsme i jiné druhy pf-diagramů a jedním z nich je typ *sinus-průsečík*. V takovém diagramu je bod reprezentující n -tý paprsek popsán:

$$X_n = [\sin \alpha_n, p_n(c)]$$

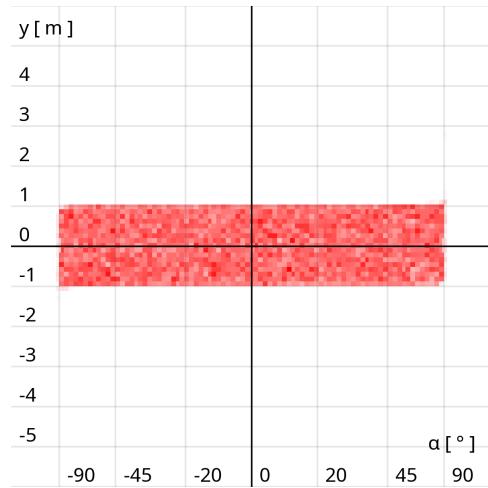
Z důvodu přehlednosti nebudeme osu x takového diagramu popisovat jako bezjednotkový $\sin \alpha$, namísto toho použijeme úhel α ve stupních ($^\circ$), který daným hodnotám sinu odpovídá. Ovšem, abychom dosáhli lineárního rozprostření hodnot $\sin \alpha$ po ose x , odpovídající hodnoty úhlu α lineárně rozprostřené nebudou. Budeme tedy de facto používat nelineární měřítko.

Rozdílů mezi zmíněnými druhy zobrazení není mnogo, ale některé přeci jen zmíníme. Jednak samozřejmě nebude platit, že při posouvání detekční přímky dochází ke zkosení obrazce, tady se budeme muset omezit na obecnější termín deformace. To je pa-

trné z následujících obrázků:

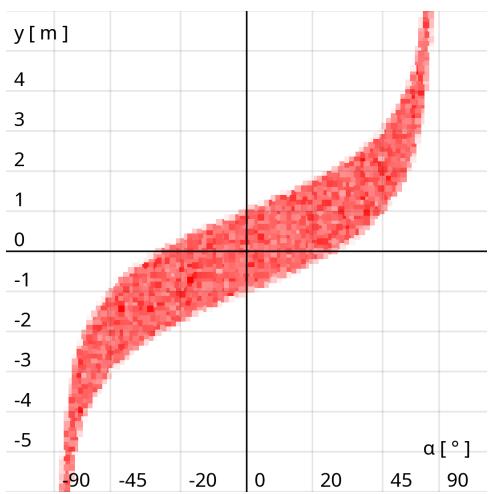


Obrázek 24: Úsečkový zdroj světla s několika znázorněnými paprsky (červená) a tří vyznačené detekční přímky, pro $x = 0$ m, $x = 2$ m a $x = 4$ m (zelená)

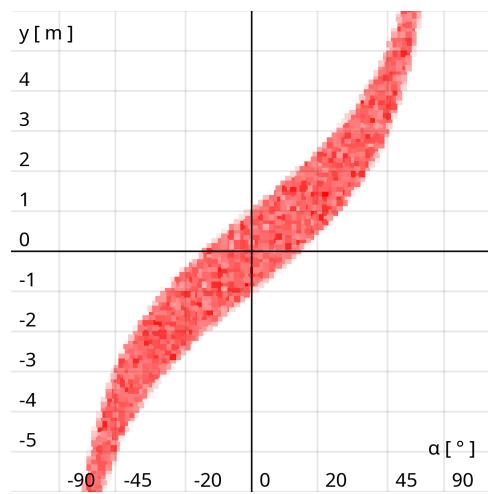


Obrázek 25: Vizualizace pf-diagramu sinus-průsečík pro levou detekční přímku $x = 0$ m v předchozím obrázku, procházející zdrojem světla

Pro situaci, kdy detekční přímka protíná zdroj, se obrazec nijak výrazně nezměnil. Rozdíl je ovšem zřejmý pro ostatní pozice detekční přímky. Závislost mezi vzdáleností bodu od osy y a rychlostí pohybu ve směru osy y již není lineární, proto se na okrajích vytváří jakési špičky.



Obrázek 26: Vizualizace pf-diagramu sinus-průsečík pro prostřední detekční přímku $x = 2$ m v předchozím obrázku, znázorňuje deformaci obrazce



Obrázek 27: Vizualizace pf-diagramu sinus-průsečík pro pravou detekční přímku $x = 4$ m v předchozím obrázku, znázorňuje deformaci obrazce

Dále si můžeme všimnout, že v tomto zobrazení se paprsky zdají být rovnoměrně rozložené, tj. diagram má všude stejný odstín červené barvy (srov. obr. 18, 19). Nezapomeňme ovšem, že stále používáme *lambertovské zářiče*. Tento způsob zakreslení pf-diagramu tedy jistým způsobem působí proti nerovnoměrnému rozložení paprsků vyžárených z úsečkového zdroje.

Dalším specifikem tohoto typu pf-diagramu je tato vlastnost:

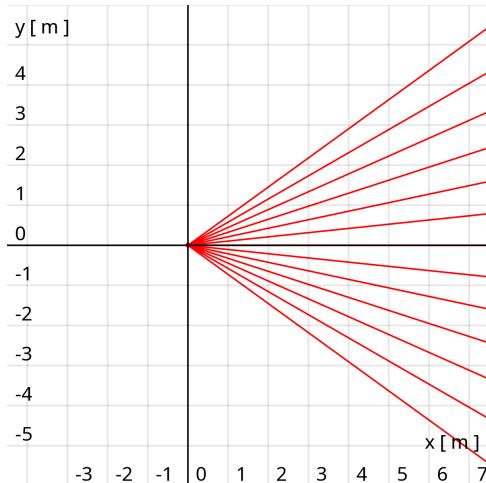
Pokud platí, že při interakci s optickým povrchem se zachovává počet paprsků (což neplatí např. pro absorpci, nedokonalý odraz apod.), je plocha obrazce tvořeného všemi paprsky konstantní, nezávisle na volbě počátku detekčního souřadného systému (v našem případě na poloze zvolené detekční přímky).

Zmíněná plocha obrazce v pf-diagramu typu *sinus-průsečík* se nazývá *étendue*. A výše popsaná vlastnost se jmenuje *zákon zachování étendue* (anglicky *conservation of etendue*). (Chaves, 2017; Muschaweck et al., 2022)

Étendue může být definováno i jako jistý součin prostorového a úhlového rozsahu. Pro svazek paprsků diferenciální velikosti (reprezentováno jedním pixelem v našem diagramu) je tento součin konstantní (všechny pixely mají stejnou plochu). Jelikož éten-due je pouze součtem těchto součinů, platí tato vlastnost i pro všechny paprsky jako celek.

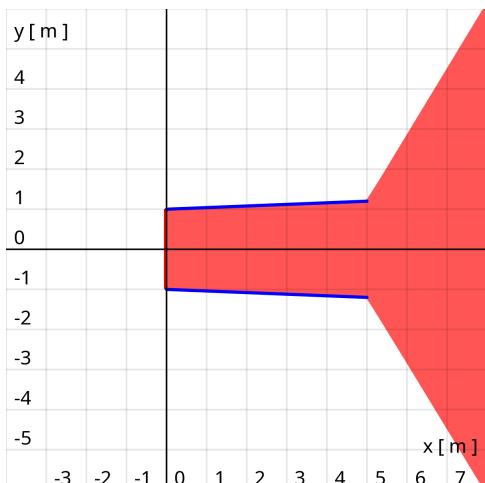
Např. na obrázku 28 je vidět, že přímo u zdroje jsou sice velké rozdíly ve sklonu paprsků (vysoký úhlový rozsah), ale všechny paprsky jsou velmi blízko u sebe (nízký prostorový rozsah). Pokud se posuneme dál od zářiče, paprsky se rozprostřou a prostorový rozsah se zvětší, avšak lokálně je úhlový rozsah v těchto místech minimální. Étendue je tedy zachováno.

S touto znalostí můžeme odvodit základní informace pro některé optické systémy. Jestliže je součin úhlového a prostorového rozsahu konstantní, lze odvodit, že úhlový rozsah je nepřímo úměrný rozsahu prostorovému.

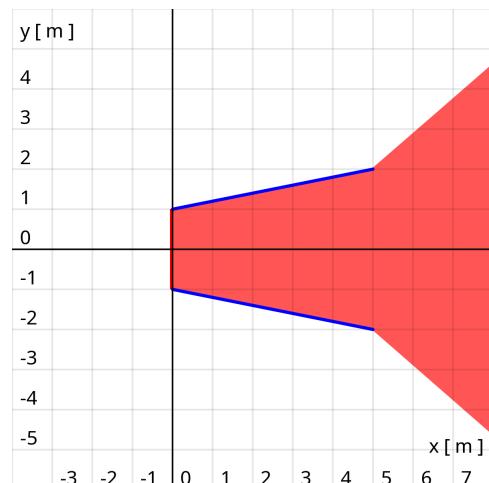


Obrázek 28: Vějířový zdroj světla s několika znázorněnými paprsky (červená)

Na příkladu již několikrát zmíněného světlovodu (z kapitoly 2.1 Vymezení problému) to znamená, že čím větší je výstupní otvor světlovodu, tím menší lokální rozptyl paprsků budeme pozorovat. To je částečně patrné i na následujících obrázcích:



Obrázek 29: Jednoduchý 2D světlovod s užším hrdlem tvořený dvěma zrcadly (modrá) s úsečkovým zdrojem světla a znázorněným světelným kuželem (červená)



Obrázek 30: Jednoduchý 2D světlovod s širším hrdlem tvořený dvěma zrcadly (modrá) s úsečkovým zdrojem světla a znázorněným světelným kuželem (červená)

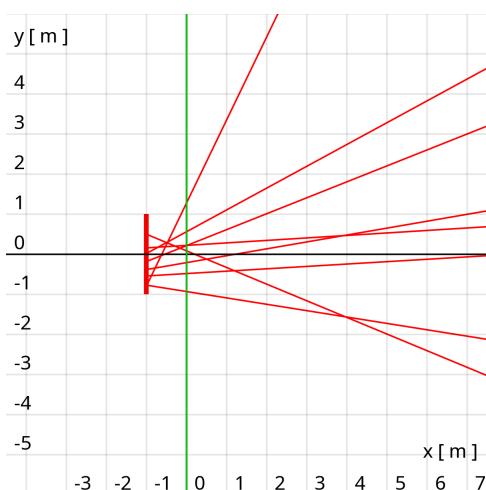
Pokud ovšem chceme za světlovod umístit další optický člen, je užitečné znát nejen globální chování světla, ale i specifické detaily pro paprsky vystupující uprostřed nebo naopak na okrajích hrdla. K tomu by měl pomoci právě pf-diagram.

2.2.6 Graf rozložení svítivosti

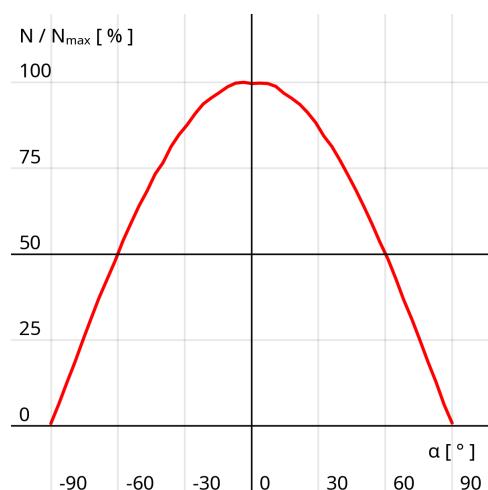
Doplňujícím typem diagramu, který se často v oblasti nezobrazovací optiky používá, je *graf rozložení svítivosti*. Ten ve zkratce ukazuje, jak velká část celkového množství paprsků míří daným směrem (neboli pod daným úhlem). Přičemž se opět zabýváme paprsky v okolí *detekční přímky*. Je ovšem nutné zdůraznit, že svítivost ($[I] = 1 \text{ cd}$) je základní *fotometrická veličina* (*Fotometrie a radiometrie*, 2006) (tj. přihlížející k lišící se citlivosti lidského oka na různé vlnové délky světla), proto je možné toto zjednodušení zavést pouze pokud předpokládáme, že všechny paprsky nesou stejný světelný tok.

Graf rozložení svítivosti lze také chápat jako jisté zjednodušení pf-diagramu. Pokud bychom sečetli počty paprsků pro jednotlivé hodnoty na ose k (vyjadřující úhel) a zakreslili je do grafu, dostali bychom právě graf svítivosti. Obecně se pf-diagram zabývá jak směrem, tak i pozicí paprsku, naproti tomu graf rozložení svítivosti se zaměřuje pouze na směr. Je proto vhodný při studiu optického soustavy z velké vzdálenosti, kdy jsou rozměry celého systému zanedbatelné.

Vodorovná osa grafu je popsána úhlem α , pod kterým se paprsky šíří. Na svislou osu budeme nanášet poměr počtu paprsků pro tento úhel vůči maximálnímu napočítanému počtu paprsků (N_α/N_{max}), v procentech (%). Příklad takového grafu je na těchto obrázcích:



Obrázek 31: Úsečkový zdroj světla s několika znázorněnými paprsky (červená) s detekční přímkou pro $x = 0$ m (zelená)



Obrázek 32: Graf rozložení svítivosti pro předchozí obrázek

Je též vhodné poukázat na fakt, že rozložení svítivosti se mění pouze pokud je některý z paprsků odražen zrcadlem nebo pohlcen clonou. Tedy např. v systému na obrázku výše by byl pro libovolnou detekční přímku napravo od zdroje ($x = c$, $c \geq -1$ m) vytvořen identický graf rozložení svítivosti.

2.3 Implementační prostředky

Vzhledem k zejména demonstrativní funkci vytvářené aplikace, se při výběru implementačních prostředků přihlíželo ke dvěma hlavním faktorům.

Jednak hrála roli vhodnost daného prostředku pro samotný úkol. Např. programovací jazyk hojně používaný na serverech se nemusí hodit pro vytváření uživatelských aplikací.

Za druhé pak rozhodovala také předpokládaná jednoduchost a rychlosť implementace z pohledu autora práce. Tzn. bylo vhodné používat prostředky, se kterými se již setkal, a nemusel se tak kromě samotné implementace ještě seznamovat s novým prostředím.

2.3.1 Programovací jazyk

Pro vývoj aplikace byl zvolen programovací jazyk *Java*. (viz java.com) Jedním z důvodů je i to, že autor práce s tímto programovacím jazykem už v minulosti pracoval.

Dalším bodem je, že standardní *SDK* (zkr. *software development kit*, volně česky *standardní knihovna*) tohoto jazyka obsahuje poměrně dobře popsaně *API* (zkr. *application programming interface*, volně česky *aplikační rozhraní*), které programům umožňuje jednoduché vytváření *GUI* (zkr. *graphical user interface*, česky *grafické uživatelské rozhraní*). To zahrnuje zobrazování oken v grafickém rozhraní operačního systému počítače a různé způsoby jak definovat a vykreslovat jejich obsah. Zmíněné *API* se v *Javě* nazývá *Java Swing*.

Pro produkční software by byla samozřejmě důležitá především rychlosť simulace. Z tohoto hlediska by bylo vhodnější použít např. programovací jazyky *C++* nebo *Rust*, které jsou na rozdíl od *Javy* komplikované přímo do strojového kódu, a proto jsou schopny jednoduché aritmetické výpočty provádět podstatně rychleji.

2.3.2 Vývojové prostředí

Vývoj aplikace probíhal v prostředí *IntelliJ IDEA* od české firmy *JetBrains*. (jetbrains.com/idea) Pro komplikaci a správu závislostí (anglicky *dependency management*, tj. především externí knihovny) byl použit program *Maven*. Důvodem byla čistě jen autova znalost těchto nástrojů.

2.4 Architektura a funkce aplikace

V této kapitole nejprve obecně popíšeme postup použití aplikace a poté rozebereme jednotlivé součásti programu.

Zmiňme, že celý zdrojový kód programu je uveden v příloze č. 1 této práce. Pro potřeby této práce používáme termín *třída* (anglicky *class*) spíše pro Javovský pojem *typ* (anglicky *type*). Ten zahrnuje jednak samotné *class*, ale i *interface* a *enum*.

2.4.1 Typické použití aplikace

Začneme tím, že uživatel definuje scénu, kterou chce prozkoumat, a následně spustí samotnou aplikaci.

V podstatě ihned poté se otevře hlavní okno aplikace zobrazující všechny prvky scény a zároveň se *na pozadí* (tzn. uživatel nevidí žádnou indikaci a nemá žádnou přímou zpětnou vazbu) zahájí i simulace průchodu paprsků scénou (anglicky *ray tracing*). Během této fáze (zatímco probíhá simulace) je možné a vhodné zkontrolovat, zda-li skutečně zadaná a zobrazená scéna odpovídá záměru.

V momentě kdy aplikace dokončí simulaci paprsků, dokreslí je do okna se scénou. Kromě nich ale i na pozici odpovídající $x = 0$ zobrazí svislou zelenou čáru, která určuje polohu *detekční přímky*. Touto čárou lze posouvat doprava a doleva s pomocí myši.

V ten samý okamžik se otevřou ještě další dvě okna, jedno obsahující *prostorovo-fázový diagram* a druhé *graf rozložení svítivosti*. Vizualizace těchto grafů reagují na změny pozice *detekční přímky* v okně se scénou.

2.4.2 Zadání scény

Z důvodu jednoduchosti implementace a snížení komplexity vývoje se scéna do aplikace zadává přímo jako kód, tj. pomocí funkcí definujících jednotlivé součásti optického systému. Když uživatel nadefinuje scénu, je nutné program zkompilovat a spustit, což by se mohlo zdát jako složitý postup, ale např. v autorem použitém prostředí IntelliJ IDEA k tomu v dnešní době stačí stisknutí jednoho tlačítka.

Pro spuštění aplikace je nutné zavolat funkci `EtendueApp.run()`, která vyžaduje jedený argument, a to objekt popisující simulovanou scénu. Právě třída `EtendueApp` propojuje

všechny části aplikace, ať se týkají simulace nebo zobrazení. Podrobněji je popisujeme dále v této kapitole.

K definici optického systému využíváme třídu Scene, což je v podstatě jen množina členů simulovaného optického systému, které jsou popsány třídou Member. V kódu definiujeme scénu pomocí funkce Scene.create(), jejíž argumenty jsou instance výše zmíněné třídy Member:

```
// Vytvoření scény
Scene scene = Scene.create(
    // Zde budou definovány jednotlivé členy této scény
);

// Samotné spuštění simulace
EtendueApp.run(scene);
```

Jednotlivým implementacím třídy Member, neboli de facto podporovaným druhům optických členů, se věnujeme podrobněji v kapitole 2.5 Druhy členů scény.

Doplňme, že v jazyce Java je potřeba pro úspěšné spuštění programu doplnit ještě několik dalších neměnných instrukcí. Pro úplnost je v tomto případě uvedeme, ale v dalších výňatcích obsahujících definice scén je vynecháme:

```
// Obsah souboru Main.java

// Tady vynecháváme ještě tzv. import statements,
// které se mění v závislosti na použitých třídách

public class Main {
    public static void main(String[] args) {

        // Vytvoření scény
        Scene scene = Scene.create(
            // Definice jednotlivých členů
        );

        // Samotné spuštění simulace
        EtendueApp.run(scene);
    }
}
```

Např. jednoduchý světlovod, zmíněný v kapitole 2.1 Vymezení problému, byl definován takto:

```
Scene scene = Scene.create(
    // Horní část
    Interactors.reflecting(
        Geometries.line(point(0, 1), point(5, 2))
    ),
    // Spodní část
    Interactors.reflecting(
        Geometries.line(point(0, -1), point(5, -2))
    ),
    // Zdroj světla
    Emitters.line(point(0, 0), 2, 10)
);

EtendueApp.run(scene);
```

S touto soustavou pracujeme i v následujících podkapitolách.

2.4.3 Simulace paprsků

Simulace průchodu paprsků scénou se provede pouze jednou, hned po spuštění programu. Výsledek je uložen do paměti pro pozdější použití, tedy zejména výpočet průsečíků s detekční přímkou.

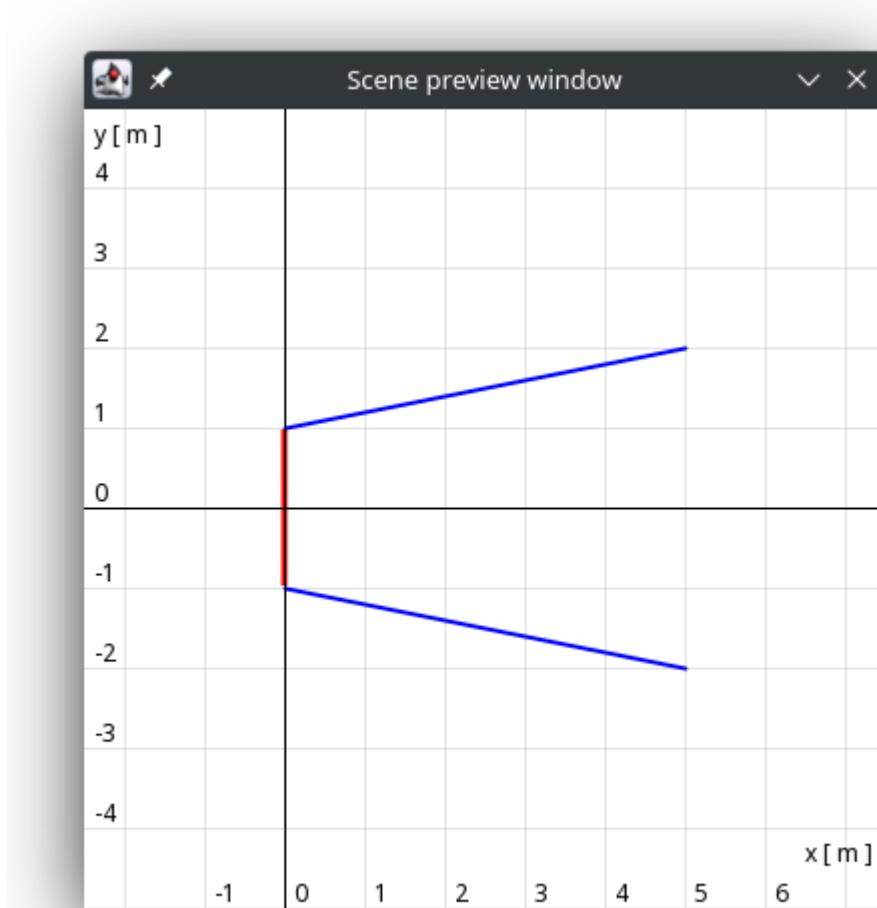
V naší aplikaci implementujeme jednoduchý *ray tracing* algoritmus. Nejprve se vysle paprsek ze zdroje světla. Poté se spočítá jeho průsečík se všemi objekty ve scéně a vybere se ten nejbližší. Následně se vyhodnotí interakce s tímto objektem (odraz nebo absorpce). A pokud se paprsek odráží, postup se opakuje. Sledování paprsku pokračuje, dokud se nedosáhne limitu počtu interakcí (v našem případě 1000), nebo dokud paprsek neopustí scénu, tzn. žádný další průsečík neexistuje.

Zajímavostí je, že pro výpočet průsečíku používáme tzv. *homogenní souřadnice* bodu a vzorec pro průsečík dvou přímk definovaný sérií vektorových součinů, jak popsal Skala (2008). Pro získaný bod poté jen ověříme, že se nachází na správné polopřímce resp. úsečce.

Dále můžeme uvést, že v zájmu zkrácení celkového času potřebného k výpočtu používáme tzv. *multithreading*, tj. způsob jak spustit více výpočtů paralelně.

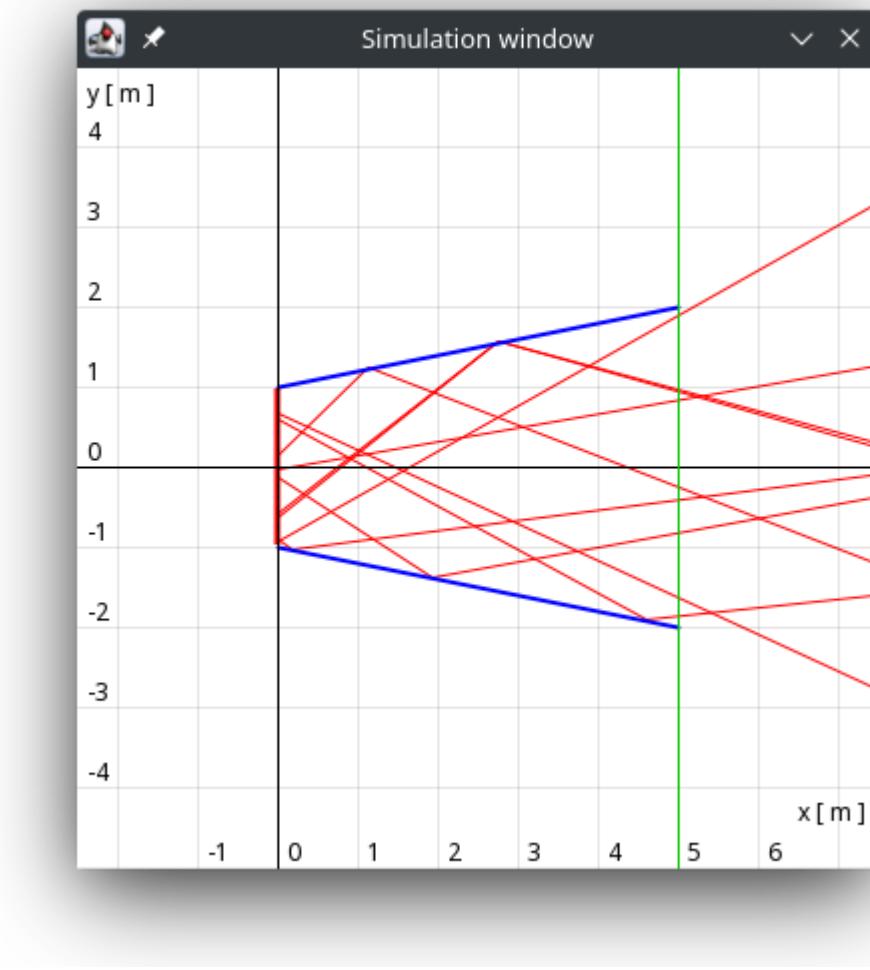
2.4.4 Vizualizace scény

Vizualizace scény proběhne za celou dobu běhu programu pouze dvakrát. Nejprve při spuštění, to jsou zobrazeny pouze jednotlivé členy scény.



Obrázek 33: Příklad okna schématu scény, ještě bez nasimulovaných paprsků

Podruhé jsou vykresleny opět všechny členy scény, ale tentokrát společně s několika náhodně vybranými paprsky (zobrazování všech simulovaných paprsků by bylo nepřehledné). Tato vizualizace se uloží do paměti programu. Když uživatel posune detekční přímku, do okna se scénou už vykreslí jen tento předvygenerovaný obrázek (nemusíme kreslit znova každý paprsek) a navrh se pouze dokreslí svislá čára znázorňující detekční přímku.



Obrázek 34: Příklad okna schématu scény, s nasimulovanými paprsky a detekční přímkou

Pro zobrazení scény jsme zvolili následující barvy: červená pro zdroje a paprsky, modrá pro zrcadla a tmavě šedá pro clony. Dále zelenou barvou je vykreslena detekční přímka a černými a světle šedými čárami je vyznačena vztažná soustava.

Zároveň uveděme, že program sám přizpůsobuje, která část scény se zobrazí a i v jakém měřítku, a to tak, aby všechny prvky scény byly vždy kompletně viditelné. Dále mřížka a její popisky se taktéž automaticky uzpůsobí zvolenému přiblžení.

2.4.5 Detekce paprsků

Výpočet průsečíků paprsků s detekční přímkou proběhne pokaždé, když se změní její poloha. Přičemž k výpočtu jsou užita uložená data o chodu paprsků z fáze sledování paprsků. Používáme stejný vzorec jako v podkapitole 2.4.3 Simulace paprsků. Jen mírně zjednodušený, protože víme, že jedna z přímek bude vždy rovnoběžná s osou y .

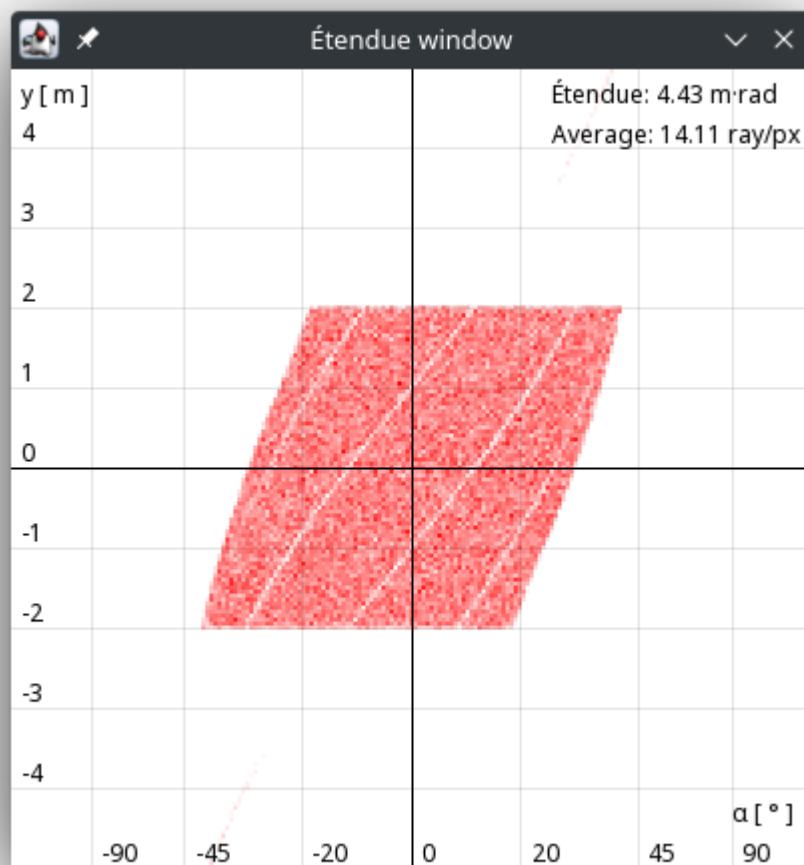
Z důvodu zajištění plynulého chodu aplikace z pohledu uživatele, probíhá i tento výpočet *na pozadí*.

2.4.6 Vizualizace prostorovo-fázového diagramu

Ihned po vypočtení průsečíků paprsků s detekční přímkou se spustí část zobrazující prostorovo-fázový diagram.

Nejprve si plochu okna pf-diagramu rozdělí na mřížku. A poté pro každý paprsek vypočte, do kterého políčka spadá. Nakonec pro každé políčko vykreslí čtvereček vybarvený odstínem červené odpovídající počtu paprsků v daném políčku.

V tomtéž okně se také zobrazí další dvě veličiny. Jednak étendue, které se vypočte jako plocha neprázdných políček pf-diagramu. A pak průměrný počet paprsků v jednom políčku (položka *Average*).



Obrázek 35: Příklad okna prostorovo-fázového diagramu

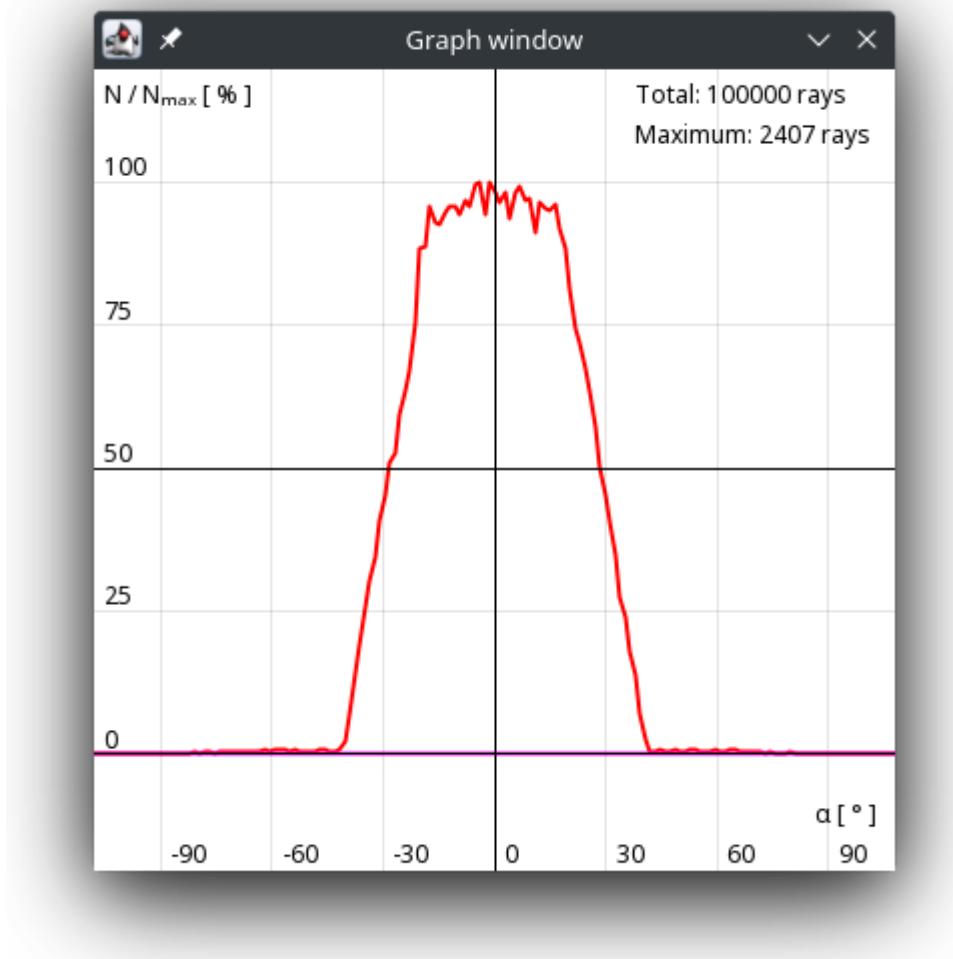
2.4.7 Vizualizace grafu rozložení svítivosti

Současně se zobrazením pf-diagramu se spustí i vykreslení grafu rozložení svítivosti. Postup je obdobný jako vizualizace pf-diagramu.

Opět se nejprve vytvoří datová struktura jako podklad pro vykreslení grafu, ovšem tentokrát je pouze jednorozměrná. Pro každý paprsek se vypočte, do kterého intervalu spadá. Na základě toho pak se zobrazí čára spojující vypočtené hodnoty.

Poznamenejme, že v tomto grafu zobrazujeme čáry dvě, jednu červenou a jednu fialovou. Červená znázorňuje počty paprsků mířící ve směru $+x$ a fialová směr $-x$.

Dále v okně vypíšeme ještě informaci o celkovém počtu paprsků (položka *Total*). A také maximální hodnotu počtu paprsků v jednom intervalu (položka *Maximum*).



Obrázek 36: Příklad okna grafu rozložení svítivosti

2.5 Druhy členů scény

V této kapitole se věnujeme jednotlivým podporovaným členům scény, tedy de facto implementacím třídy Member (viz kapitola 2.4.2 Zadání scény).

Tyto implementace můžeme rozdělit do dvou kategorií, na zdroje světla (třída Emitter) a na opticky interagující prvky (třída Interactor).

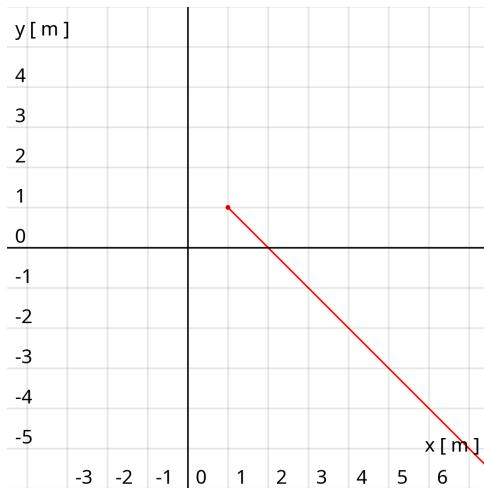
2.5.1 Světelné zdroje

Jediným úkolem instance třídy Emitter je vytvořit jakýsi seznam paprsků, které daný objekt vysílá. V našem případě jde o množinu dvojic *počáteční bod – směrový vektor*.

V aplikaci je definováno několik základních druhů světelých zdrojů. Se všemi z nich jsme se již v této práci setkali.

Jednoduchý zdroj je definován bodem a vektorem. Vysílá právě jeden paprsek.

```
// Funkce vytvářející jednoduchý zdroj
Emitters.single(
    // Počáteční bod
    point(1, 1),
    // Směrový vektor
    vector(1, -1)
)
```



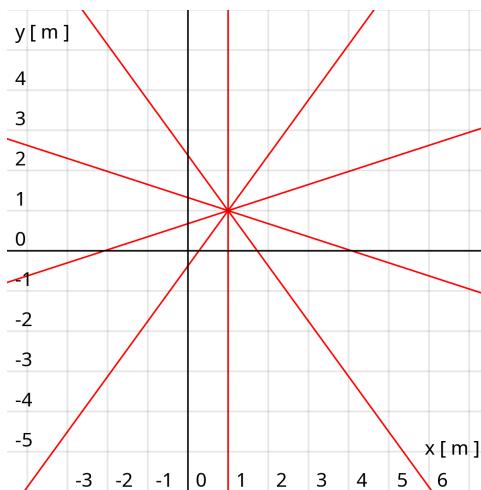
Obrázek 37: Příklad scény obsahující pouze jednoduchý zdroj světla

Kromě již popsaných částí kódu si všimněme použití pomocných funkcí point a vector, které vytvářejí objekty reprezentující bod resp. vektor (v obou případech jde de facto

o uspořádanou dvojici čísel). Tyto funkce se hodí zejména pro lepší čitelnost a jednodušší porozumění definice z pohledu uživatele.

Bodový zdroj je definován bodem a počtem paprsků. Vysílá daný počet paprsků rovnoměrně rozložený do všech směrů.

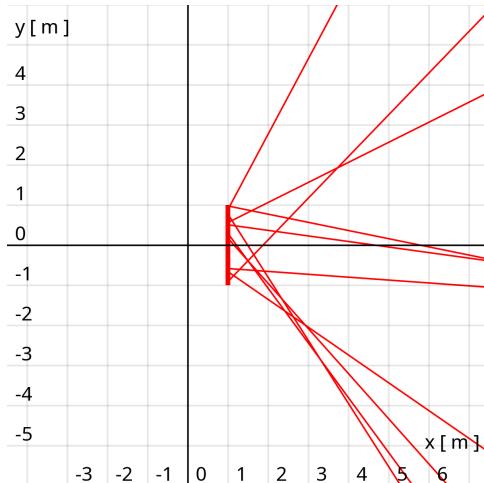
```
// Funkce vytvářející bodový zdroj
Emitters.point(
    // Střed zdroje
    point(1, 1),
    // Počet paprsků
    10
)
```



Obrázek 38: Příklad scény obsahující pouze bodový zdroj světla

Úsečkový zdroj je definován středem, délkou a počtem paprsků. Je vždy orientován svisle a paprsky vysílá vždy směrem doprava. O vlastnostech tohoto typu zářiče jsme se již zmiňovali v kapitole 2.2.4 Vlastnosti prostorovo-fázového diagramu, ale pro jistotu zopakujme, že jej považujeme za *lambertovský zářič*. (*Fotometrie a radiometrie*, 2006)

```
// Funkce vytvářející úsečkový zdroj
Emitters.line(
    // Střed zdroje
    point(1, 0),
    // Délka zářiče
    2,
    // Počet paprsků
    10
)
```



Obrázek 39: Příklad scény obsahující pouze úsečkový zdroj světla

Uživateli je samozřejmě umožněno definovat vlastní světelné zářiče. Např. vějířový zdroj (viz obr. 28) je velmi jednoduchou úpravou bodového zdroje.

2.5.2 Typy optických prvků

Obecně instance třídy `Interactor` nesou dvě informace. Jednak samozřejmě svůj tvar, a pak také nějaký předpis, který říká, co se děje s dopadajícími paprsky.

Jak je vysvětleno již v kapitole 2.1 Vymezení problému, v této aplikaci jsme se omezili pouze na dva typy optických členů, *dokonalé clony* a *dokonalá zrcadla*. Funkce, které je vytvářejí, vyžadují pouze jeden argument, a tím je objekt popisující tvar. Různým vyjádřením tvaru se venujeme v následující podkapitole.

Dokonalá clona se vytváří takto:

```
// Funkce vytvářející dokonalou clonu
Interactors.absorbing(
    // Zde je definice tvaru
)
```

Dokonalé zrcadlo se vytváří následujícím způsobem:

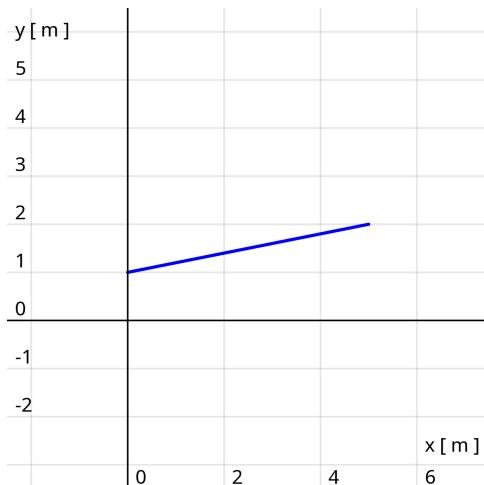
```
// Funkce vytvářející dokonalé zrcadlo
Interactors.reflecting(
    // Zde je definice tvaru
)
```

2.5.3 Geometrie optických prvků

Aplikace podporuje několik různých typů geometrií optických prvků. My se zde zmíníme o dvou.

Úsečka je definována pomocí dvou bodů:

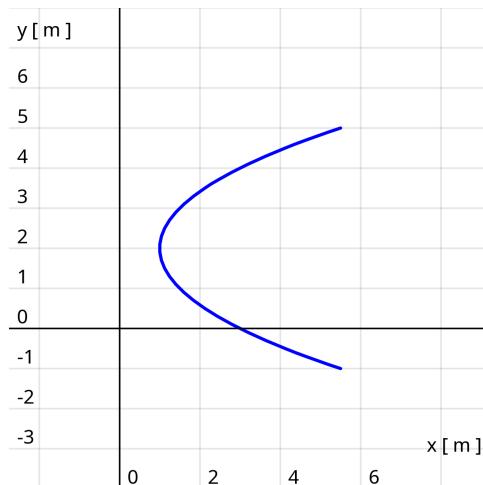
```
// Funkce vytvářející úsečkovou geometrii
Geometries.line(
    // Počáteční bod
    point(0,1),
    // Koncový bod
    point(5,2)
)
```



Obrázek 40: Příklad scény obsahující úsečkovou geometrii

Geometrie podle matematické funkce je zadávána poněkud složitěji. Parametry jsou popsány přímo v následujícím kódu.

```
// Funkce vytvářející geometrii podle funkce
Geometries.formula(
    // Vyjadřujeme x v závislosti na y, nebo obráceně
    true,
    // Počáteční bod, vyjadřuje, kde se promítne bod [0,0]
    point(1, 2),
    // Výchozí hodnota parametru
    -3,
    // Koncová hodnota parametru
    3,
    // Krok parametru
    0.1f,
    // Samotná funkce, do které je dosazován parametr
    x -> x*x / 2
)
```



Obrázek 41: Příklad scény obsahující approximaci paraboly

3 Výsledky

Vybrali jsme celkem pět různých optických systémů, které je možné rozdělit do dvou kategorií, *světlovody* a *parabolické reflektory*. Pro každý z těchto soustav jsme spustili náš program a poté uložili vypočtené vizualizace a grafy.

3.1 Světlovody

3.1.1 Krátký světlovod

První systém je zadán takto:

Jednoduchý zrcadlový světlovod se šírkou vstupu 2 m a šírkou výstupu 5 m.

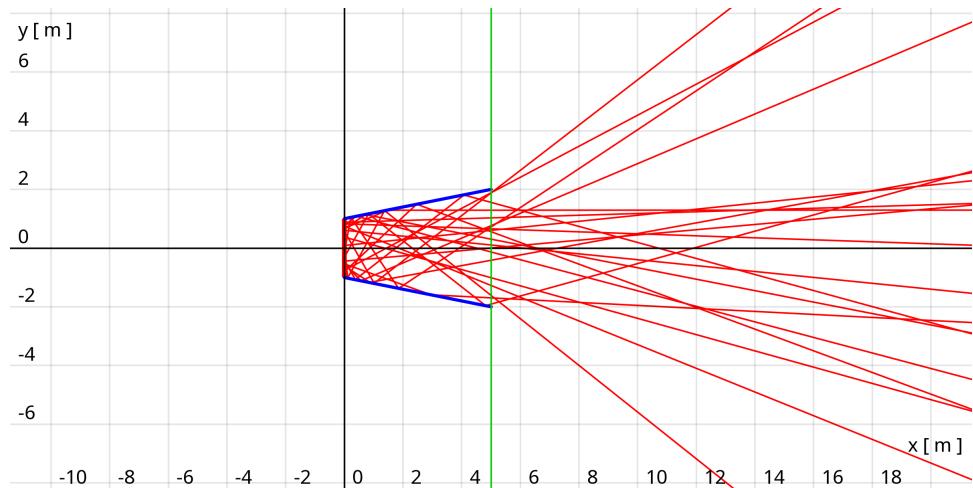
Jeho celková délka činí 5 m a na vstupu do světlovodu je připevněn úsečkový zdroj o délce 1,95 m.

Odpovídající definice scény:

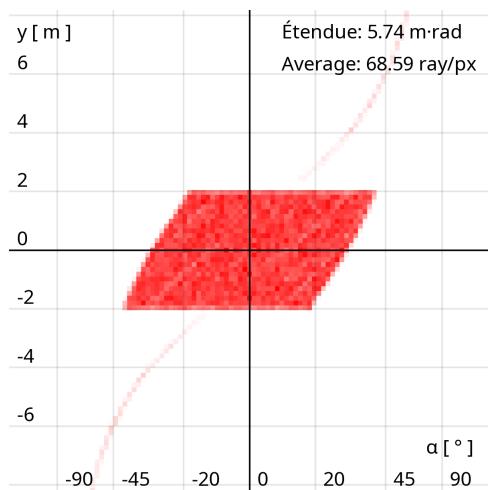
```
Scene scene = Scene.create(
    // Horní část světlovodu
    Interactors.reflecting(
        Geometries.line(point(0, 1), point(5, 2))
    ),
    // Spodní část světlovodu
    Interactors.reflecting(
        Geometries.line(point(0, -1), point(5, -2))
    ),

    // Zdroj paprsků
    Emitters.line(
        point(0, 0),
        1.95f,
        100000
    )
);
EtendueApp.run(scene);
```

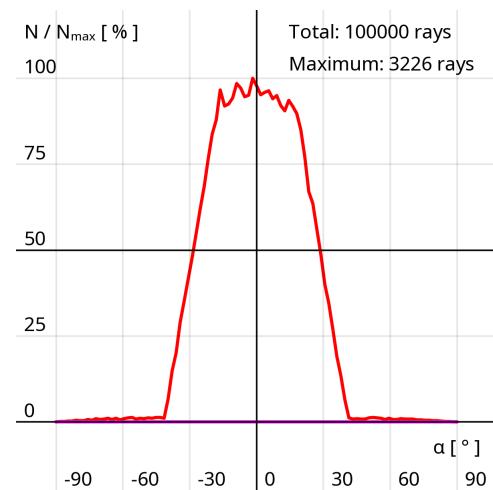
Vygenerované vizualizace:



Obrázek 42: Schéma krátkého světlovodu



Obrázek 43: Pf-diagram na výstupu z krátkého světlovodu



Obrázek 44: Rozložení svítivosti po průchodu krátkým světlovodem

3.1.2 Dlouhý světlovod

Druhý systém, taktéž světlovod, se od prvního liší pouze v celkové délce, která je dvojnásobná:

Jednoduchý zrcadlový světlovod se šírkou vstupu 2 m a šírkou výstupu 5 m.

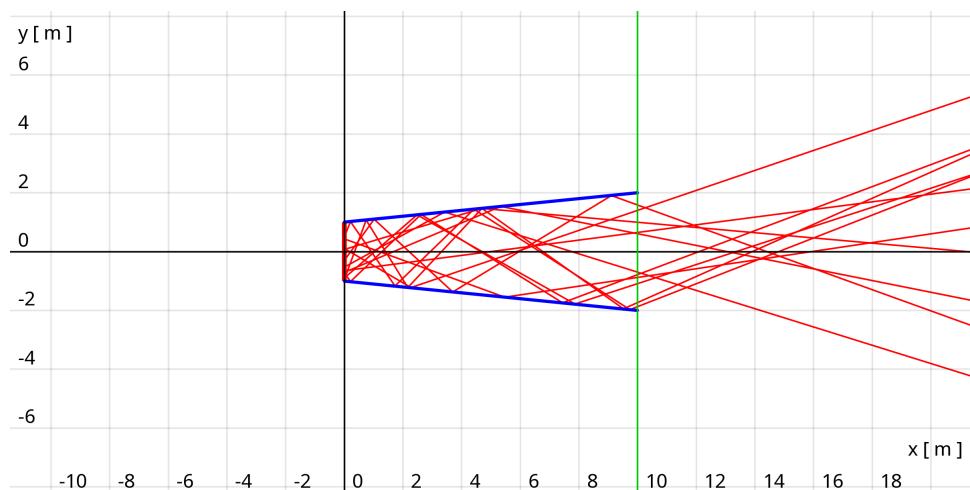
Jeho celková délka činí 10 m a na vstupu do světlovodu je připevněn úsečkový zdroj o délce 1,95 m.

Odpovídající definice scény:

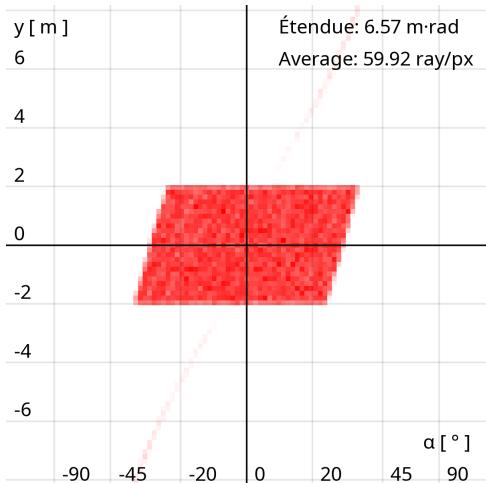
```
Scene scene = Scene.create(
    // Horní část světlovodu
    Interactors.reflecting(
        Geometries.line(point(0, 1), point(10, 2))
    ),
    // Spodní část světlovodu
    Interactors.reflecting(
        Geometries.line(point(0, -1), point(10, -2))
    ),

    // Zdroj paprsků
    Emitters.line(
        point(0, 0),
        1.95f,
        100000
    )
);
EtendueApp.run(scene);
```

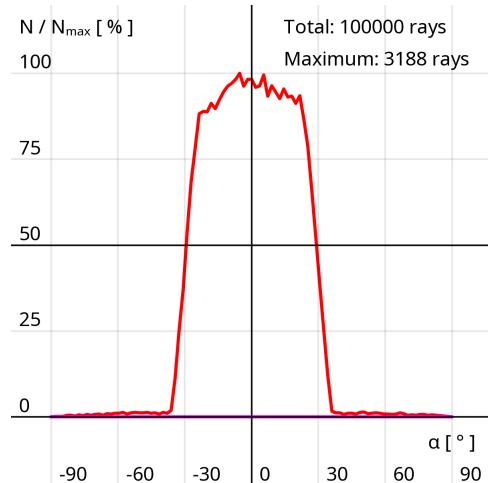
Vygenerované vizualizace:



Obrázek 45: Schéma dlouhého světlovodu



Obrázek 46: Pf-diagram na výstupu z dlouhého světlovodu



Obrázek 47: Rozložení svítivosti po průchodu dlouhým světlovodem

3.2 Parabolické reflektory

3.2.1 Parabolický reflektor s ohniskem ve středu vstupního otvoru

Pokračujeme do kategorie parabolických reflektorů. První takový systém je popsán takto:

Parabolický reflektor se vstupem šířky 2 m a délkou 5 m. Parabola je umístěna tak, aby její ohnisko leželo na středu vstupního otvoru. Ve něm je také připevněn úsečkový zdroj o délce 1 m.

Tuto scénu jsme definovali následujícím způsobem. Poukažme na to, že jsou použity proměnné za účelem zjednodušení případné změny v zadání:

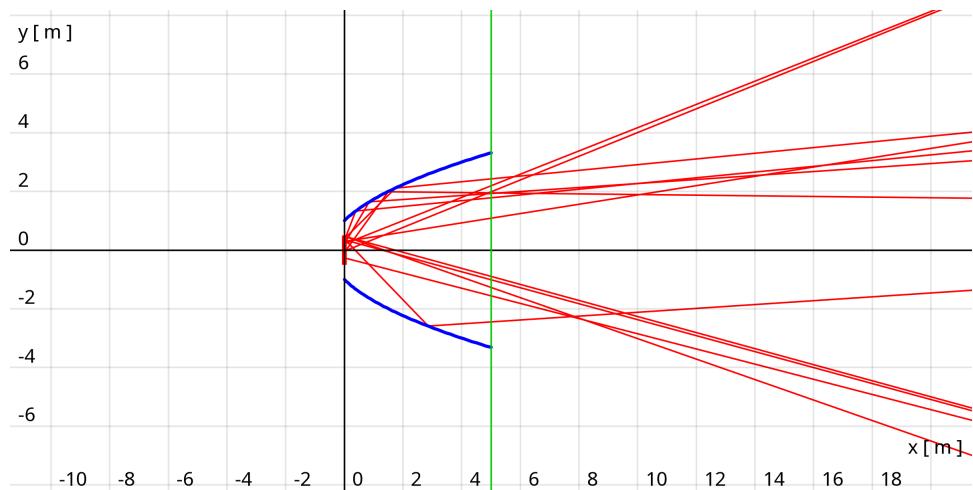
```
// Délka reflektoru
float length = 5f;
// Poměr vzdáleností ohniska paraboly a vstupu reflektoru
// od vrcholu paraboly
float offsetRatio = 1f;

// Obecné vyjádření reflektoru pro výše zadané rozměry.
// Předpoklad je, že vstupní otvor se nachází na souřadnici
// x = 0 m a má velikost 2 m.

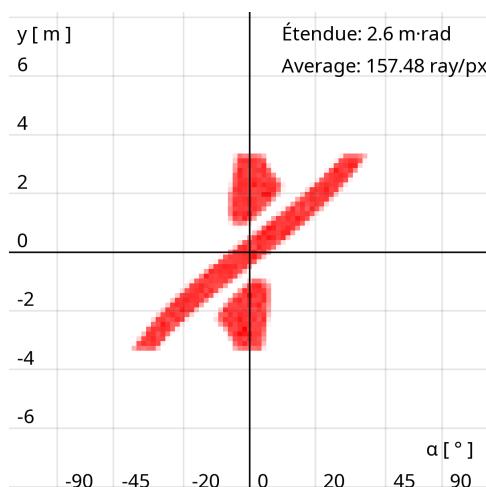
// Vypočtený parametr obecné rovnice paraboly
float param = (float) Math.sqrt(offsetRatio);
// Posunutí po ose x, aby reflektor začínal na x = 0
float offset = param / (2 * offsetRatio);
// Maximální hodnota, pro kterou počítáme tvar reflektoru
float max = (float) Math.sqrt( (length+offset) * 2*param );

Scene scene = Scene.create(
    // Horní část reflektoru
    Interactors.reflecting(
        Geometries.formula(
            true,
            point(-offset, 0),
            1, max,
            0.05f, x -> x*x / (2*param)
        )
    ),
    // Spodní část reflektoru
    Interactors.reflecting(
        Geometries.formula(
            true,
            point(-offset, 0),
            -max, -1,
            0.05f, x -> x*x / (2*param)
        )
    ),
    // Zdroj paprsků
    Emitters.line(
        point(0, 0),
        1f,
        100000
    )
);
EtendueApp.run(scene);
```

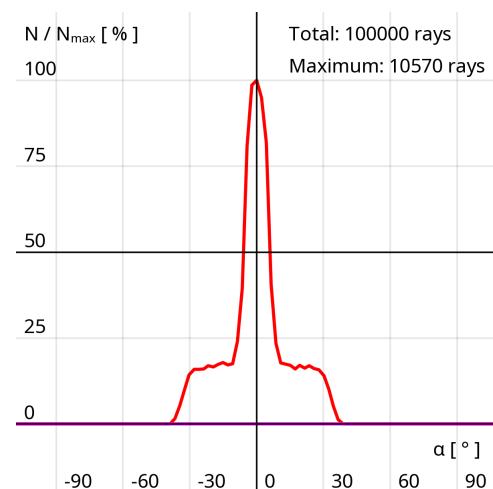
Vygenerované vizualizace:



Obrázek 48: Schéma parabolického reflektoru s ohniskem ve středu vstupního otvoru



Obrázek 49: Pf-diagram na výstupu z parabolického reflektoru s ohniskem ve středu vstupního otvoru



Obrázek 50: Rozložení svítivosti po průchodu parabolickým reflektorem s ohniskem ve středu vstupního otvoru

3.2.2 Parabolický reflektor s ohniskem za vstupním otvorem (uvnitř těla reflektoru)

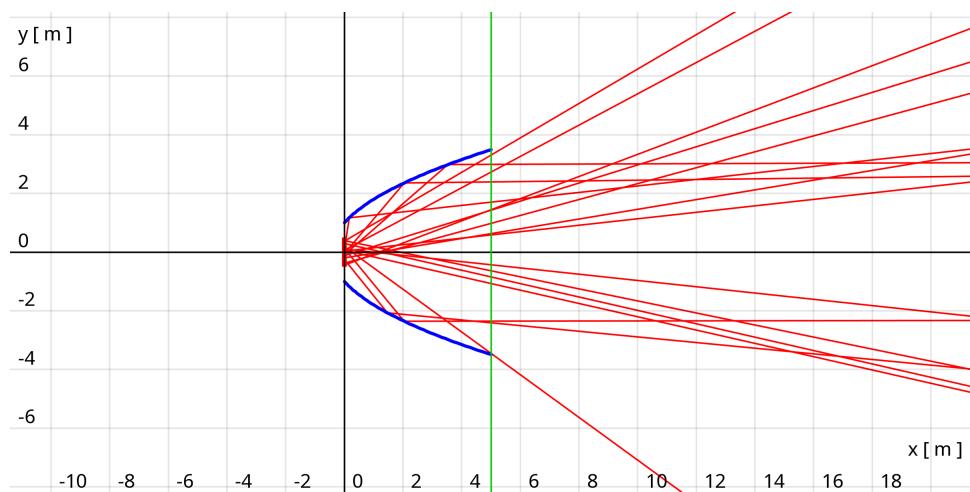
Dalším systémem je podobný parabolický reflektor jako v předchozím případě, ohnisko paraboly je jen posunuto mírně doprava, zatímco ostatní podmínky zůstávají stejné:

Parabolický reflektor se vstupem šířky 2 m a délku 5 m. Parabola je umístěna tak, aby její ohnisko leželo uvnitř těla reflektoru. Ve vstupním otvoru je připevněn úsečkový zdroj o délce 1 m.

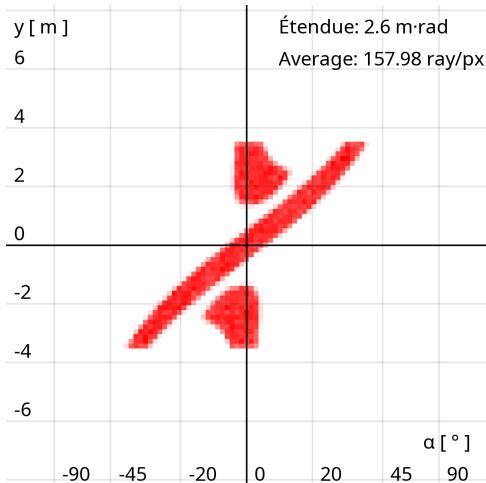
Zdůrazněme, že v tomto kontextu nemá posun ohniska za následek posun celé paraboly, nýbrž v tomto případě je nutné celou rovnici paraboly přepočítat. Zde se nám využije výše zmíněná definice pomocí několika proměnných, díky ní stačí změnit pouze jedno číslo:

```
// Délka reflektoru  
float length = 5f;  
// Ohnisko uvnitř reflektoru  
float offsetRatio = 7 / 5f;  
  
// ...  
// Zbytek kódu je stejný
```

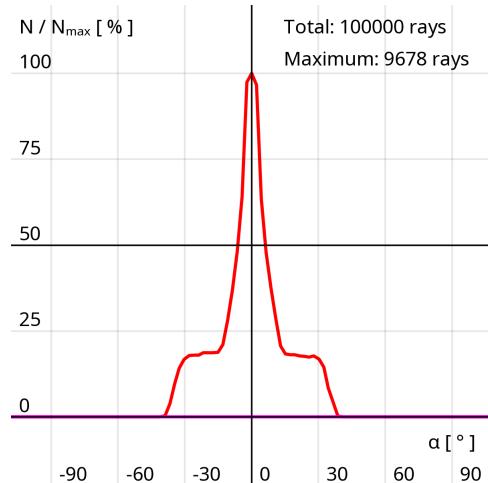
Vygenerované vizualizace:



Obrázek 51: Schéma parabolického reflektoru s ohniskem za vstupním otvorem



Obrázek 52: Pf-diagram na výstupu z parabolického reflektoru s ohniskem za vstupním otvorem



Obrázek 53: Rozložení svítivosti po průchodu parabolickým reflektorem s ohniskem za vstupním otvorem

3.2.3 Parabolický reflektor s ohniskem před vstupním otvorem (mimo tělo reflektoru)

Podobná úprava reflektoru jako v předchozím případě, ovšem ohnisko je posunuto doleva:

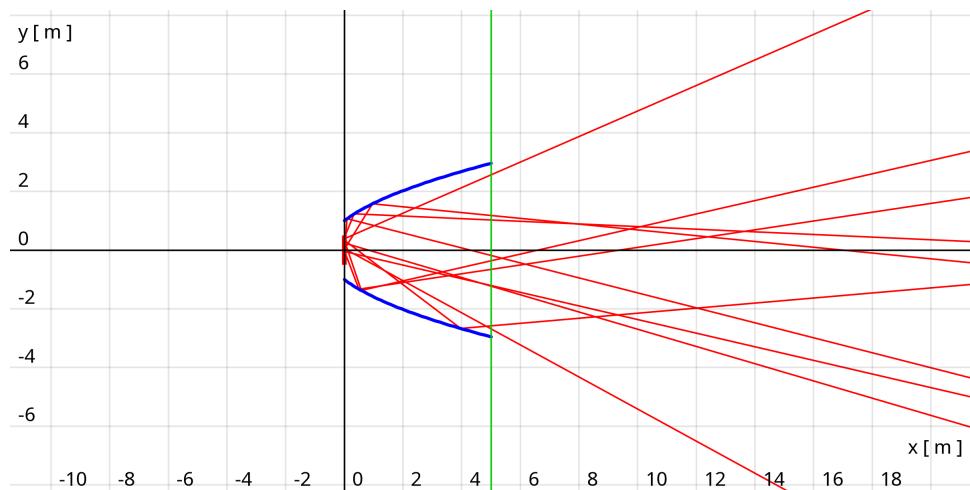
Parabolický reflektor se vstupem šířky 2 m a délkou 5 m. Parabola je umístěna tak, aby její ohnisko leželo nalevo od těla reflektoru. Ve vstupním otvoru je připevněn úsečkový zdroj o délce 1 m.

Odpovídající definice scény:

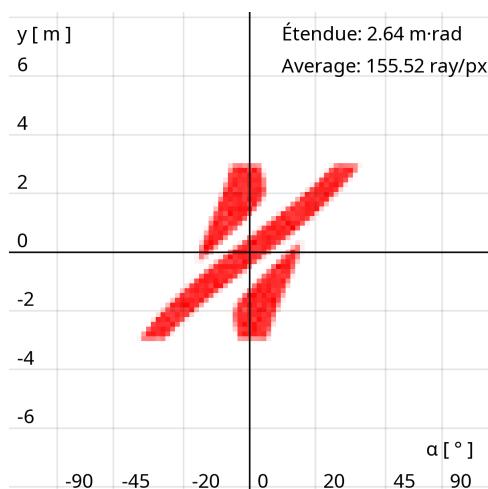
```
// Délka reflektoru
float length = 5f;
// Ohnisko mimo reflektor
float offsetRatio = 3 / 4f;

// ...
// Zbytek kódu je stejný
```

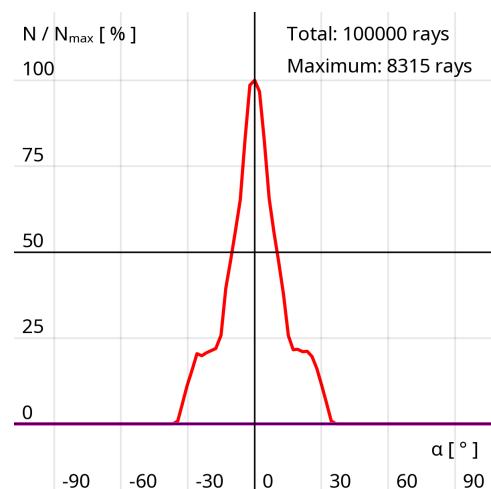
Vygenerované vizualizace:



Obrázek 54: Schéma parabolického reflektoru s ohniskem před vstupním otvorem



Obrázek 55: Pf-diagram na výstupu z parabolického reflektoru s ohniskem před vstupním otvorem



Obrázek 56: Rozložení svítivosti po průchodu parabolickým reflektorem s ohniskem před vstupním otvorem

4 Diskuze

4.1 Celková užitečnost

Celkově můžeme říct, že se aplikace prokázala jako užitečná. Příkladem nám poslouží simulace z kapitoly 3.2 Parabolické reflektory. Předpokládejme, že naším cílem je soustředit co nejvíce světla ve směru osy paraboly. Pokud porovnáme pf-diagramy pro různé pozice ohnisek, je zřejmé, že sám o sobě je pro nás nejlepší reflektor se zdrojem v ohnisku (obr. 48). A to protože centra odražených paprsků jsou přímo uprostřed pf-diagramu, tedy mají sklon blízký nule.

Ovšem pokud bychom měli možnost napravo od reflektoru umístit ještě zobecněnou (*freeformovou*) čočku, z pf-diagramu zjistíme, že nejlepších výsledků dosáhneme pokud bude ohnisko posunuté doprava, neboli dovnitř reflektoru (obr. 51). A to z toho důvodu, že útvary odražených paprsků se posunou na stranu směrem ke zbytku neodražených paprsků. Takovýto tvar může zobecněná čočka posunout zpět směrem k nulovému sklonu. Nejenže vrátí centra odražených paprsků zpět na nulový sklon (jak tomu bylo u předchozího příkladu), ale společně s nimi zlomí i všechny neodražené paprsky blíže k ose reflektoru. Tudíž ve výsledku celý systém soustřídí více světla ve směru osy x . Takováto konfigurace by tedy stála za další prozkoumání.

Na užitečnosti vytvořené aplikace se shodlo i několik odborníků zabývajících se návrhem nezobrazovací optiky. (P. Lobaz, osobní komunikace, 24. 2. 2023)

Dále můžeme také pozorovat, že vytvořená aplikace má i několik nedostatků, které by se daly rozdělit do dvou hlavních kategorií. Vzhledem k cíli této práce, kterýmžto je pouze ověřit použitelnost konceptu, jím však nepřikládáme velkou váhu.

4.2 Nedostatky návrhu aplikace

Tyto problémy se týkají samotného konceptu aplikace.

Prvním mírným nedostatkem je samozřejmě již několikrát zmiňovaná absence pokročilejších funkcí, tj. složitější geometrie, dalsí typy interakcí apod. (viz kapitola 2.1 Vy mezení problému). Ovšem aplikace je navržena tak, že uživatel se základní znalostí programování je schopen bez většího zásahu potřebné možnosti doplnit.

Dalším problémem je nutnost definice scény přímo v kódu a v nikoli nějakém exter-

ním souboru (podkapitola 2.4.2 Zadání scény), jako tomu často bývá u podobných simulacních aplikací. To má za následek nutnost kompilovat program při každém spuštění a vyžaduje po uživateli minimálně základní znalost programování. V ideálním případě by náš program měl podporovat alespoň některé hojně používané formáty v oblasti simulace optiky.

V souvislosti s tím bylo též vhodné umožnit uživateli kromě změny pozice detekční přímky i úpravu scény přímo v běžící aplikaci, aby se opět nemusel celý program pouštět znova.

4.3 Nedostatky implementace aplikace

Tato kategorie se zabývá výpočetními chybami, které se týkají až hotové aplikace. Mohou být způsobeny buďto limitacemi přímo programovacího jazyka *Java* nebo neoptimálním řešením samotného autora aplikace.

První z chyb je dobře viditelná na některých vizualizacích pf-diagramu. Specificky to jsou obr. 43 a 46. Nad i pod hlavní částí obrazce jsou viditelné nezřetelné čáry. Všimněme si, že podle zadání obou příkladů v související kapitole 3.1 Světlovody se celý zdroj světla nachází uvnitř reflektoru, ještě s mírnou rezervou. Ovšem paprsky, které jsou zmíněnými čárami znázorněny se zcela jistě nacházejí mimo světlovod. Jediné vysvětlení je, že při sledování chodu tyto paprsky nějakým způsobem prošly skrze zrcadlo, aniž by se odrazily.

To je způsobeno kombinací několika faktorů. Tím prvním je čistě způsob implementace ray tracing procesu (podkapitola 2.4.3 Simulace paprsků). Při něm totiž po nalezení dalšího průsečíku program kontroluje, jestli není příliš blízko poslednímu známému bodu. To má za cíl zabránit dvojitěmu odrazu (tzn. paprsek se odrazí dvakrát od toho stejného zrcadla na stejném místě a vypadá to, jako kdyby jen prošel skrz). Dalo by se říci, že použitý způsob jak zabránit chybám v simulaci sám vytvoří další chyby. Ovšem tento nový druh chyb je podstatně méně četný a vyskytuje se jen ve specifických případech. Druhým důvodem jsou nepřesnosti ve výpočtech s tzv. floating-point číselnými formáty. („IEEE Standard for Floating-Point Arithmetic“, 2008) Kvůli nim nemůžeme v při sledování chodu paprsků jednoduše porovnat jestli poslední známý bod je shodný s právě nalezeným průsečíkem. To protože v mnoha případech není, i když

čísla se liší jen o velmi malé zlomky. Proto se musíme uchýlit ke kontrolování vzdálostí těchto dvou bodů.

Druhá chyba implementace programu se tentokrát týká výpočtu étendue. Ze *zákona zachování étendue* (kapitola 2.2 Analýza systému) vyplývá, že pro jeden a ten samý zdroj, nehledě na okolní optické prvky a volbu detekční přímky, bude étendue konstantní. Ovšem, jak je vidět na příkladech se světlovodem (obr. 43 a 46), ale i s parabolickým reflektorem (obr. 49, 52 a 55), námi vypočtené étendue tomu nevždy odpovídá.

Nepřesnost těchto hodnot je způsobena zvoleným způsobem výpočtu. Étendue počítáme jako plochu pf-diagramu a ten zobrazujeme po ve zvolené mřížce. Pro lepší kvalitu výpočtu by bylo vhodné snížit velikost jednotlivých políček mřížky. Ovšem to by vyžadovalo zvýšení počtu simulovaných paprsků, aby se zamezilo případům, že nějaké políčko zcela jasně uvnitř obrazce nebude zasaženo žádným paprskem. Simulace vyššího množství paprsků by zase trvala déle, proto v tomto případě bylo nutné najít kompromis mezi rychlostí a přesností výpočtu.

5 Závěr

V rámci této maturitní práce jsem se zabýval problematikou návrhu nezobrazovacích optických systémů. Specificky jsem ověřoval užitečnost vizualizace prostorovo-fázového diagramu pro 2D systémy.

Vytvořil jsem počítačovou aplikaci, která dokáže sledovat chod paprsků 2D scénou složenou ze zdrojů světla a optických prvků (clona a zrcadlo) vyjádřených úsečkami. Dále jsem implementoval vizualizaci jak samotné scény, tak i průchodu jednotlivých paprsků. Vytvořený program umí samozřejmě zobrazit zmíněný prostorovo-fázový diagram a také vypočítat jeho plochu, zvanou étendue. Poslední implementovanou částí aplikace je funkce vykreslení grafu rozložení svítivosti.

Společně s odborníkem z praxe jsem došel k závěru, že tento typ aplikace se zdá být užitečný. Bylo by proto vhodné se touto problematiku dále zabývat, atž už ve formě dalšího vylepšování simulační aplikace, nebo uskutečnění podobného rozboru užitečnosti i pro 3D systémy.

Seznam použité literatury

- Fotometrie a radiometrie, 2006. Praha: Fyzikální ústav Univerzity Karlovy. Dostupné také z: http://fu.mff.cuni.cz/biomolecules/media/files/courses/Fotometrie_a_radiometrie.pdf. [cit. 2023-02-28].
- HANNAVY, John, 2008. *Encyclopedia of nineteenth century photography*. London: Routledge. ISBN 978-0415972352.
- CHAVES, Julio, 2017. *Introduction to nonimaging optics, second edition*. 2. vyd. London: CRC Press. ISBN 978-1482206746.
- IEEE Standard for Floating-Point Arithmetic*, 2008. IEEE Std 754.
- MUSCHAWECK, Julius A; REHN, Henning, 2022. *Designing Illumination Optics*. Bellingham: SPIE Press. Tutorial Texts.
- SKALA, Václav, 2008. Intersection computation in projective space using homogeneous coordinates. *International Journal of Image and Graphics*. Roč. 8, č. 4, s. 615–628. ISBN 978-1510649354.
- WIMMER, Wolfgang, 2017. Carl Zeiss, Ernst Abbe, and Advances in the Light Microscope. *Microscopy Today*. Roč. 25, č. 4, s. 50–57.
- ŽÁRA, Jiří; BENEŠ, Bedřich; SOCHOR, Jiří; FELKEL, Petr, 2005. *Moderní počítačová grafika*. 2. vyd. Praha: Computer Press. ISBN 80-251-0454-0.

Seznam obrázků

1	Jednoduchý 2D světlovod tvořený dvěma zrcadly (modrá) s úsečkovým zdrojem světla a několika znázorněnými paprsky (červená)	9
2	Příklad dokonalého odrazu na zrcadle (modrá) s bodovým zdrojem světla a několika znázorněnými paprsky (červená)	11
3	Příklad dokonalé absorpce na cloně (šedá) s bodovým zdrojem světla a několika znázorněnými paprsky (červená)	11
4	Aproximace parabolického zrcadla (modrá) s úsečkovým zdrojem světla a několika znázorněnými paprsky (červená)	12
5	Jednoduchý 2D světlovod tvořený dvěma zrcadly (modrá) s úsečkovým zdrojem světla a několika znázorněnými paprsky (červená), zakreslený v systému Oxy (černá a šedá)	14
6	Scéna s několika barevně označenými paprsky	15
7	Vizualizace paprsků z předchozího obrázku v pf-diagramu	15
8	Scéna s několika barevně označenými paprsky a detekční přímkou (zelená) pro $x = 1 \text{ m}$	16
9	Vizualizace paprsků z předchozího obrázku v pf-diagramu pro detekční přímku $x = 1 \text{ m}$	16
10	Vějířový zdroj světla s několika znázorněnými paprsky (červená) a tři vyznačené detekční přímky, pro $x = 0 \text{ m}$, $x = 2 \text{ m}$ a $x = 4 \text{ m}$ (zelená)	17
11	Vizualizace pf-diagramu pro levou detekční přímku $x = 0 \text{ m}$ v předchozím obrázku, procházející zdrojem světla	17
12	Vizualizace pf-diagramu pro prostřední detekční přímku $x = 2 \text{ m}$ v předchozím obrázku, znázorňuje zkosení obrazce	17
13	Vizualizace pf-diagramu pro pravou detekční přímku $x = 4 \text{ m}$ v předchozím obrázku, znázorňuje zkosení obrazce	17
14	Zdroj rozvnoběžného svazku paprsků, z nichž některé jsou znázorněny (červená), a dvě vyznačené detekční přímky, pro $x = 0 \text{ m}$ a $x = 4 \text{ m}$ (zelená)	18
15	Vizualizace pf-diagramu pro levou detekční přímku $x = 0 \text{ m}$ v předchozím obrázku, procházející zdrojem světla	18

16	Vizualizace pf-diagramu pro pravou detekční přímku $x = 4$ m v předchozím obrázku, beze změny oproti diagramu procházejícím zdrojem světla	18
17	Úsečkový zdroj světla s několika znázorněnými paprsky (červená) a dvě vyznačené detekční přímky, pro $x = 0$ m a $x = 4$ m (zelená)	19
18	Vizualizace pf-diagramu pro levou detekční přímku $x = 0$ m v předchozím obrázku, procházející zdrojem světla	19
19	Vizualizace pf-diagramu pro pravou detekční přímku $x = 4$ m v předchozím obrázku, znázorňuje zkosení obrazce	19
20	Úsečkový zdroj světla s několika znázorněnými paprsky (červená), clona (šedá) a vyznačená detekční přímka $x = 4$ m (zelená)	20
21	Vizualizace pf-diagramu scény na předchozím obrázku	20
22	Úsečkový zdroj světla s několika znázorněnými paprsky (červená), zrcadlo (modrá) a vyznačená detekční přímka $x = 4$ m (zelená)	21
23	Vizualizace pf-diagramu scény na předchozím obrázku, obsahuje neodražené paprsky (červená) a odražené paprsky (modrá)	21
24	Úsečkový zdroj světla s několika znázorněnými paprsky (červená) a tři vyznačené detekční přímky, pro $x = 0$ m, $x = 2$ m a $x = 4$ m (zelená)	22
25	Vizualizace pf-diagramu sinus-průsečík pro levou detekční přímku $x = 0$ m v předchozím obrázku, procházející zdrojem světla	22
26	Vizualizace pf-diagramu sinus-průsečík pro prostřední detekční přímku $x = 2$ m v předchozím obrázku, znázorňuje deformaci obrazce	22
27	Vizualizace pf-diagramu sinus-průsečík pro pravou detekční přímku $x = 4$ m v předchozím obrázku, znázorňuje deformaci obrazce	22
28	Vějířový zdroj světla s několika znázorněnými paprsky (červená)	24
29	Jednoduchý 2D světlovod s užším hrdlem tvořený dvěma zrcadly (modrá) s úsečkovým zdrojem světla a znázorněným světelným kuželem (červená)	24
30	Jednoduchý 2D světlovod s širším hrdlem tvořený dvěma zrcadly (modrá) s úsečkovým zdrojem světla a znázorněným světelným kuželem (červená)	24
31	Úsečkový zdroj světla s několika znázorněnými paprsky (červená) s detekční přímkou pro $x = 0$ m (zelená)	25
32	Graf rozložení svítivosti pro předchozí obrázek	25

33	Příklad okna schématu scény, ještě bez nasimulovaných paprsků	31
34	Příklad okna schématu scény, s nasimulovanými paprsky a detekční přímou	32
35	Příklad okna prostorovo-fázového diagramu	34
36	Příklad okna grafu rozložení svítivosti	35
37	Příklad scény obsahující pouze jednoduchý zdroj světla	36
38	Příklad scény obsahující pouze bodový zdroj světla	37
39	Příklad scény obsahující pouze úsečkový zdroj světla	38
40	Příklad scény obsahující úsečkovou geometrii	39
41	Příklad scény obsahující approximaci paraboly	40
42	Schéma krátkého světlovodu	42
43	Pf-diagram na výstupu z krátkého světlovodu	42
44	Rozložení svítivosti po průchodu krátkým světlovodem	42
45	Schéma dlouhého světlovodu	43
46	Pf-diagram na výstupu z dlouhého světlovodu	44
47	Rozložení svítivosti po průchodu dlouhým světlovodem	44
48	Schéma parabolického reflektoru s ohniskem ve středu vstupního otvoru	46
49	Pf-diagram na výstupu z parabolického reflektoru s ohniskem ve středu vstupního otvoru	46
50	Rozložení svítivosti po průchodu parabolickým reflektorem s ohniskem ve středu vstupního otvoru	46
51	Schéma parabolického reflektoru s ohniskem za vstupním otvorem	47
52	Pf-diagram na výstupu z parabolického reflektoru s ohniskem za vstupním otvorem	48
53	Rozložení svítivosti po průchodu parabolickým reflektorem s ohniskem za vstupním otvorem	48
54	Schéma parabolického reflektoru s ohniskem před vstupním otvorem	49
55	Pf-diagram na výstupu z parabolického reflektoru s ohniskem před vstupním otvorem	49
56	Rozložení svítivosti po průchodu parabolickým reflektorem s ohniskem před vstupním otvorem	49