Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra informatiky a výpočetní techniky

Bakalářská práce

Grafické uživatelské rozhraní aplikace pro simulaci holografie

Plzeň 2016 Lucie Herejtová

Prohlášení

Prohlašuji, že jsem bakalářskou práci vypracovala samostatně a výhradně s použitím citovaných pramenů.

V Plzni dne 29. dubna 2016

Lucie Herejtová

Abstract

Graphical user interface of application for holography simulation

Motivation for developing an application, which simulates optical phenomenons, is a possibility to try to create complicated optical systems and become acquainted with hologram's principle, without the need to actually produce these optical systems. This bachelor thesis follows several bachelor and master theses, which were dealing with developing previously mentioned application. Main goal of this thesis is a implementation of a graphical user interface for a new version of this application, while the most important objective is adding support for displaying holograms and optical systems in 3D space.

Abstrakt

Motivací pro tvorbu aplikace, simulující optické jevy, je možnost zkusit si vytvořit složitější optické soustavy a seznámit se s principem hologramu, aniž by bylo potřeba tyto soustavy vyrábět fyzicky. Tato bakalářská práce navazuje na několik bakalářských a diplomových prací, v jejichž rámci zmíněná aplikace v minulých letech vznikala. Jejím cílem je implementace grafického rozhraní k nové verzi této aplikace, přičemž nejdůležitější bodem je rozšíření podpory zobrazení optických soustav a hologramů ve 3D prostoru.

Obsah

1 Úvod

V minulých letech, v rámci bakalářských prací [1, 2], vznikala aplikace pro vizualizaci principu hologramu. Obecnou motivací pro vytváření simulátoru je jednak možnost vyzkoušet si sestavení složitějších optických soustav, ale také slouží jako výuková aplikace pro pochopení principu hologramu. V prvotních verzích obsahovala pouze základní optické členy (světlo a záznamový materiál) a pracovala ve dvourozměrným prostoru. Později byla doplněna o další členy (rovinné zrcátko, tenkou čočku a dělič svazku).

Novým cílem je aplikaci rozšířit o podporu zobrazení ve 3D prostoru, přičemž tento úkol byl rozdělen na dvě části. První z nich, návrh a implementace výpočetního jádra, byla realizována Kamilem Rendlem v rámci jeho diplomové práce [3]. Druhá část, návrh a implementace grafického rozhraní, je obsahem této bakalářské práce. Kromě uživatelské intuitivnosti a přístupnosti je důraz kladen především na správné zobrazení objektů ve 3D prostoru.

Ve druhé kapitole je představen původní simulátor, ve třetí kapitole prozkoumáme dostupné programy pro paprskovou optiku a modelování scén ve 3D prostoru. Z poznatků z těchto dvou kapitol navrhneme nové grafické uživatelské rozhraní v kapitole 4. V páté kapitole si představíme výslednou aplikaci a podrobně popíšeme důležité implementační detaily. Následuje kapitola o uživatelském testování aplikace.

2 Původní verze

V této kapitole si nejprve aplikaci představíme z pohledu uživatelského rozhraní a poté si ukážeme jak se s ní pracuje při modelování konkrétní optické soustavy (Mach-Zehnderův interferometr) a vytváření a rekonstrukci jednoduchého hologramu.

2.1 Grafické uživatelské rozhraní

Po spuštění se zobrazí okno aplikace (viz obr. 2.1), které je rozděleno na tři části:

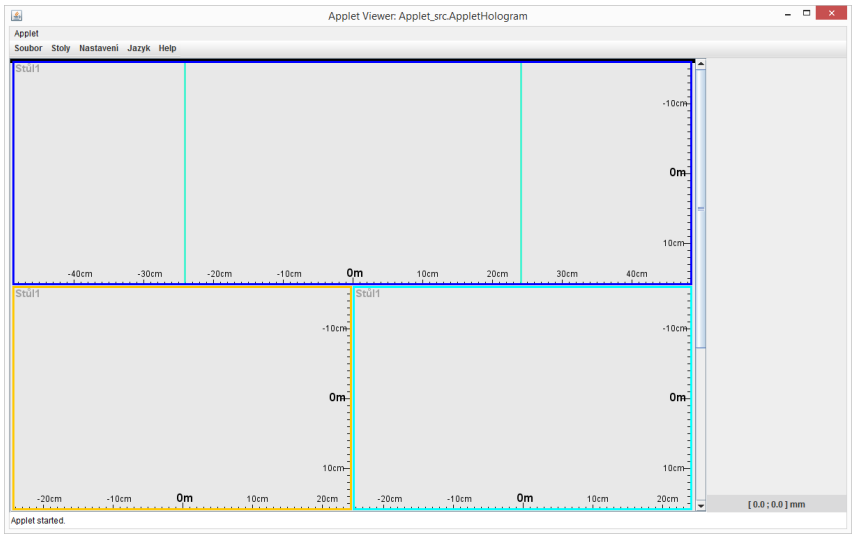
- **pracovní plocha** – Pracovní plocha (vlevo) představuje pohledy na jednotlivé `stoly`[1], na které je možné přidávat několik optických objektů: světlo, záznamový materiál, zrcátko, čočku a dělič svazku.
- **panel nastavení** – V panelu nastavení (vpravo) lze měnit parametry a nastavení vybraného optického objektu.
- **menu** – Menu umožňuje například přidávat další stoly a vytvářet tak několik modelů najednou, nastavovat obecné vlastnosti aplikace, nebo změnit jazykovou lokalizaci.

Pracovní plocha

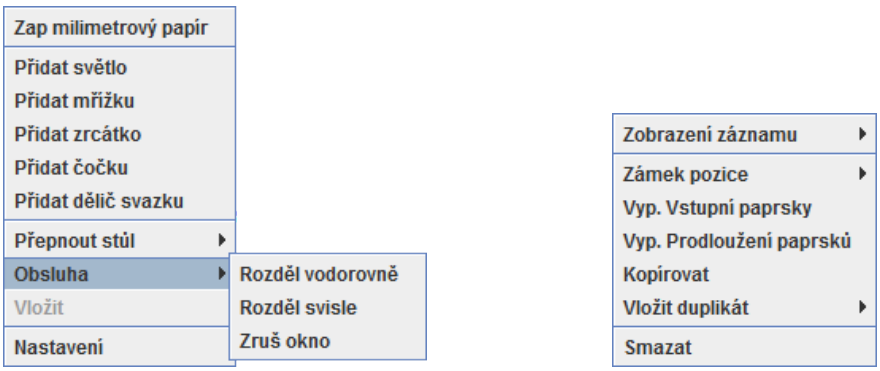
Pracovní plochu lze libovolně rozdělovat pomocí obsluhy v kontextovém menu vyvolaném pravým tlačítkem kdekoli na požadovaném `pohledu`[2] (viz obr. 2.2a). Pohledy lze pochopitelně v tomto kontextovém menu taktéž rušit – je možné mít jediný pohled přes celou pracovní plochu, vyžaduje-li optická soustava rozsáhlejší prostor.

[1] `Stůl` si lze představit jako reálný stůl, na kterém se vytváří optické soustavy a simulují hologramy.

[2] Pohledem je myšleno konkrétní zobrazení stolu ohraničené jednou barvou; na jeden stůl může být více pohledů.



Obrázek 2.1: Hlavní okno aplikace po spuštění.



(a) Obsluha stolu. (b) Kontextové menu optického členu.

Obrázek 2.2: Kontextová menu

Díky dělení pracovní plochy je umožněna práce na několika stolech současně, mezi nimiž lze pak v jednotlivých pohledech přepínat tím samým menu (obr. 2.2a) pod položkou `Přepnout stůl`. V tomto menu jsou také položky pro přidávání jednotlivých optických členů.

Užitečnou možností kontextového menu optického objektu (viz obr. 2.2b), vyvolaného klikem pravým tlačítkem myši na tento objekt, je zamknutí jedné nebo obou os kartézského souřadného systému (dále jen KSS). Díky této volbě lze pohybovat objektem pouze v jednom směru, případně zamknutím obou směrů znemožnit pohyb úplně (například chceme-li zamezit náhodnému po-

hybu objektem při práci s optickou soustavou).

Postranní panel

Panel nastavení, nacházející se v okně aplikace (obr. 2.1) vpravo, slouží k zobrazení informací o posledním vybraném objektu (vybíráme klikem levého tlačítka myši na objekt v pracovní ploše) a jeho nastavení. Lze jej vidět na obrázku 2.3.

Obrázek 2.3: Panel nastavení pro světlocitlivý materiál – mřížku.

Jak je z obrázku patrné, je panel dělen na několik částí, přičemž všechny optické členy mají společné tyto části:

- **Akce** – Obsahuje tlačítka akcí, které lze s tímto optickým členem provádět.
- **Parametry** – Zobrazuje měnitelné parametry objektu.
- **Seznam osvětlujících**[3] **prvků** – Zde lze nastavit, jakými zdroji bude

[3]Pro optický člen `Světlo` osvětlených.

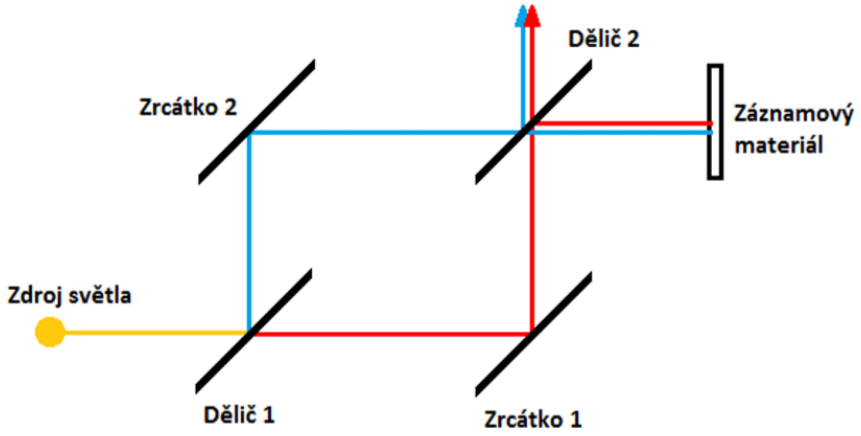
tento objekt osvětlen (pro opt. člen `Světlo` naopak – nastavuje se, jaké objekty bude osvětlovat).

Pro některé optické členy obsahuje postranní panel další části – např. pro `Světlo` je to informace o vlnové délce vyzařovaného světla a o tom, je-li světlo monochromatické, nebo polychromatické. Ve spodní části panelu také vidíme aktuální umístění (souřadnice KSS) kurzoru na pracovní ploše, což napomáhá umístění objektu na konkrétní pozici.

2.2 Práce se simulátorem

2.2.1 Modelování optické soustavy

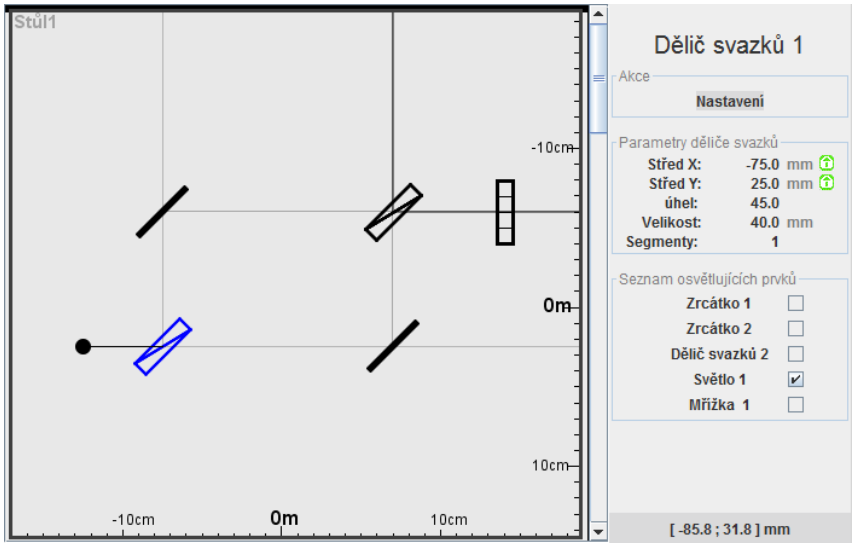
Ukažme si práci se simulátorem na Mach-Zehnderově interferometru (viz obr. 2.4). Skládá se ze světelného zdroje, dvou zrcátek, dvou děličů svazku a záznamového materiálu.



Obrázek 2.4: Mach - Zehnderův interferometr [2].

Nejprve vložíme na stůl pomocí kontextového menu pracovní plochy, vyvolaného pravým tlačítkem myši, všechny požadované optické objekty, tj. zdroj světla, dvě zrcátka, dva děliče svazku a záznamový materiál (mřížku). Dále upravíme pozice všech prvků tak, jak vidíme na obrázku 2.4. To provedeme kliknutím na objekt a táhnutím myši, nebo v části *Parametry* postranního panelu, kde můžeme každému prvku nastavit konkrétní souřadnice. Zde také postupně obě zrcátka i děliče svazku otočíme o 45°.

Nyní již stačí nastavit každému objektu, jak bude osvětlován ostatními objekty (check boxy ve spodní části panelu nastavení – viz obr. 2.3). *Dělič 1* je osvětlen *Světelným zdrojem*, obě zrcátka *Děličem svazku 1*, *Dělič svazku 2* oběma zrcátky a *Záznamový materiál* osvětlíme *Děličem svazku 2*. Výsledný model a parametry nastavené *Děliči svazku 1* můžeme vidět níže na obrázku 2.5.



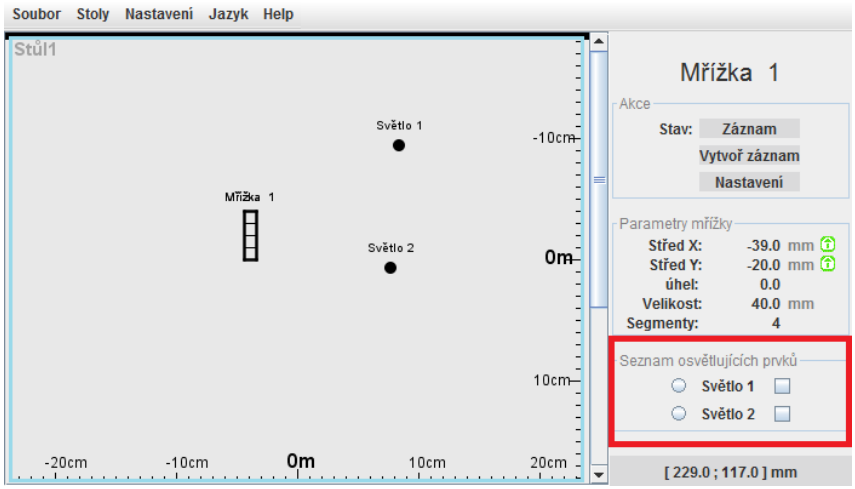
Obrázek 2.5: Mach - Zehnderův interferometr namodelovaný v simulátoru.

2.2.2 Záznam a rekonstrukce hologramu

Dále si ukážeme práci s aplikací na jedné z nejdůležitějších funkcí, ke kterým je aplikace určena – záznamu a rekonstrukci hologramu.

V první řadě musíme na stůl umístit (pomocí kontextového menu stolu – viz obr. 2.2a) objekty, potřebné k vytvoření záznamu hologramu (např. obr. 2.6). Jsou to dva bodové zdroje světla, z nichž jeden slouží jako referenční světlo a druhý jako zjednodušení zaznamenávaného objektu, a holografickou desku (v terminologii aplikace též difrakční mřížka, či pouze mřížka). Aby bylo možné hologram vytvořit, musí být obě světla umístěna na stejné straně od mřížky.

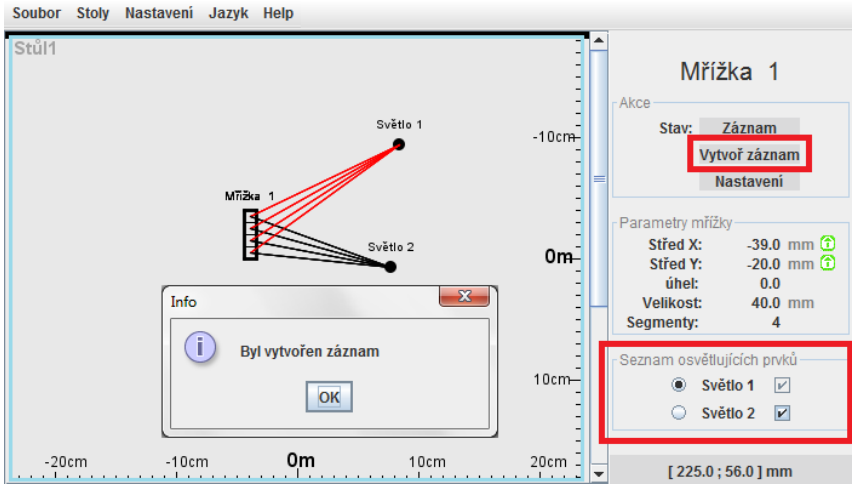
Poté nastavíme osvícení mřížky, což provedeme v postranním panelu v části *Seznam osvětlujících prvků* (viz obr. 2.6 vpravo dole). Check boxem jsou určeny



Obrázek 2.6: Možné rozmístění objektů pro vytvoření záznamu hologramu.

světelné zdroje, které budou mřížku osvětlovat; radio button pak určuje, které světlo bude referenční. Podle obr. 2.7 zaškrtneme oba check boxy a vybereme, které světlo bude referenční – zde *Světlo 1*[4].

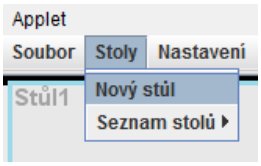
Nyní – máme-li správně nastavené osvětlení mřížky – můžeme vytvořit záznam. Tuto akci provedeme tlačítkem *Vytvoř záznam* v horní části postranního panelu (viz obr. 2.7).



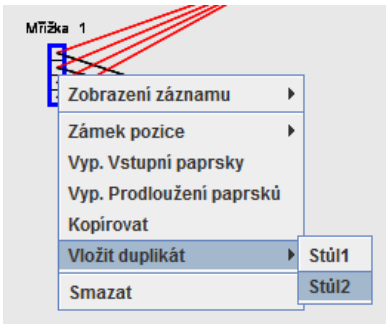
Obrázek 2.7: Nastavení osvícení mřížky a vytvoření záznamu hologramu.

[4] *Světlo 2* tedy představuje zaznamenávaný objekt.

Tímto je záznam hologramu hotov a můžeme přistoupit k rekonstrukci. Nejprve vytvoříme nový stůl (viz obr. 2.8a), na kterém budeme hologram rekonstruovat a poté na něj z prvního stolu duplikujeme (viz obr. 2.8b) záznamový materiál (*Mřížka 1*).



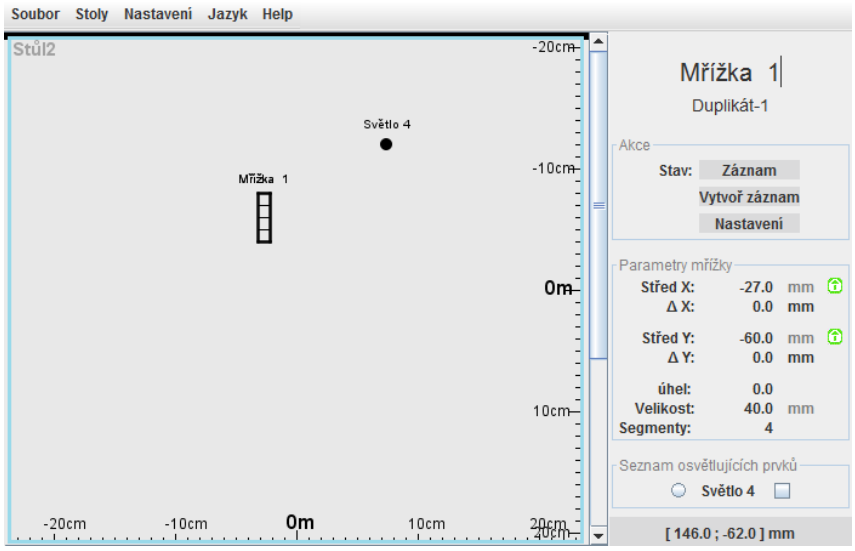
(a) Přidání stolu.



(b) Duplikace objektu.

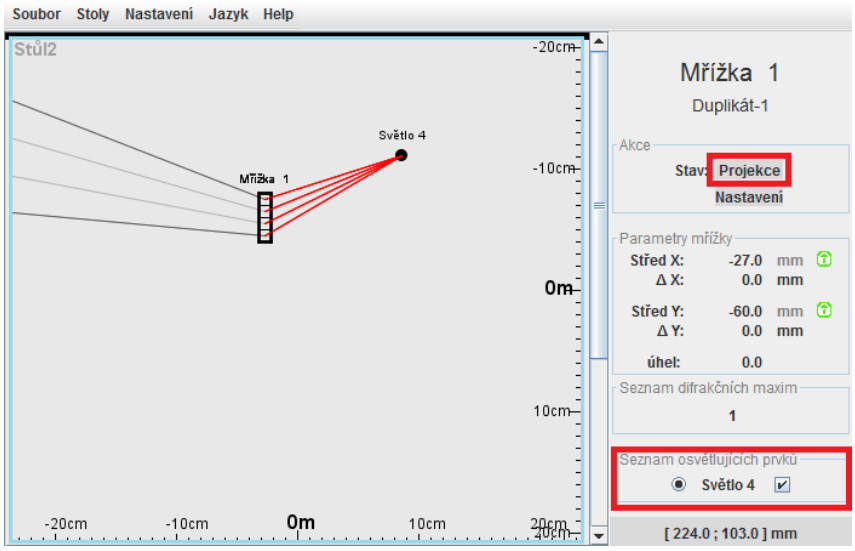
Obrázek 2.8: Přidání nového stolu a duplikace objektu

Následně se přepneme do kontextu nově vytvořeného stolu položkou *Přepnout stůl* v kontextovém menu stolu (viz obr. 2.2a), kde vidíme zduplikovanou holografickou desku. Na stůl známým způsobem přidáme světelný zdroj, který bude sloužit jako rekonstrukční světlo. Výsledek těchto akcí může vypadat např. podle obr. 2.9.



Obrázek 2.9: Stůl se záznamem hologramu a rekonstruujícím světlem.

Nyní je nutné přepnout mřížku do režimu projekce (kliknutím na tlačítko *Záznam* nahoře v panelu nastavení) a osvítit ji rekonstruujícím světlem podle obrázku 2.10. Rekonstrukce hologramu je hotova – můžeme pozorovat ohyb paprsků na difrakční mřížce.



Obrázek 2.10: Rekonstruovaný hologram.

3 Rozbor existujících aplikací

Tato kapitola se zabývá zkoumáním možností, které poskytují GUI aplikací zabývajících se podobnou problematikou, jako tato práce (jednoduché umisťování objektů do 3D prostoru, paprsková optika). Z aplikací zabývajících se 3D modelováním byly zkoumány `Blender` a `SketchUp`, z paprskové optiky se jedná o aplikace `Ray Optics Simulation` (dále také ROS) a původní verze.

Pozornost bude věnována především aspektům a vlastnostem GUI (výše zmíněných programů) využitelných v této práci, tj. umisťování objektů ve 3D prostoru, manipulace s nimi a promítání 3D scény na 2D plochu (obrazovku).

3.1 Přidávání objektů

Přidávání objektů musí být rychlé a přímočaré (například je nevhodné, aby se objekt přidával pomocí tlačítka schovaného v rozbalovací nabídce horního menu aplikace). Všechny zkoumané programy mají tlačítka pro přidávání na hlavní ploše, tzn. je možné požadovaný objekt přidat na scénu pouze dvěma kliky myši (výběr objektu a umístění). Dle mého názoru je nejvhodnější způsob přidávání objektů v aplikaci Ray Optics Simulation – vybereme objekt (viz obr. 3.1), který chceme přidávat, a poté ho lze na scénu přidávat opakovaně klikáním myši na požadované souřadnice – než je tomu například v aplikaci Blender, kde se po kliku na tlačítko přidání objektu umístí objekt tam, kde je umístěn ukazatel (který jsme dříve umístili do scény klikem myši).

Obrázek 3.1: Menu pro přidávání objektů aplikace Ray Optics Simulation s vybranou položkou *Beam*.

Nutno ovšem podotknout, že chceme-li přidat pouze jediný objekt určitého typu, je takový způsob spíše komplikací, a výhodnější je způsob, kterým objekty přidává původní verze aplikace (viz obr. 2.2a). Představuji si tedy, že z tohoto důvodu bude nová aplikace schopna přidávat nové objekty oběma způsoby.

Vhodnou funkcí je možnost objekty také kopírovat. Zde naopak poskytuje lepší způsob aplikace Blender – objekt lze vložit do schránky, odkud je možno jej vkládat vícekrát na libovolné pozice ve scéně. V aplikaci Ray Optics Simulation je nutné kliknout po vybrání požadovaného objektu na tlačítko duplikace, které umístí kopii na stejnou pozici, kde se nachází originál.

3.2 Posun s objektem

Změna pozice je jednou z nejdůležitějších, a vskutku nejvíce využívaných, akcí, které GUI nabízí, nebot' prakticky nelze namodelovat jakoukoliv soustavu objektů bez využití posunu. Je tedy nezbytné, aby byla uživateli co nejlépe přístupná a bez potíží ovladatelná.

Například v aplikaci SketchUp je posun zbytečně komplikovaný – nejprve musíme objekt vybrat (3 rychlé kliky myší), kliknout na tlačítko posunu v menu aplikace a teprve poté pohybujeme objektem posunem myši za držení jejího levého tlačítka (drag and drop). Nejjednodušší a pro uživatele nejvýhodnější postup poskytuje aplikace ROS – zde se jedná pouze o klasický drag and drop.

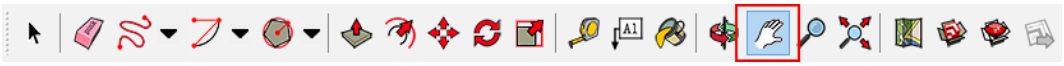
Některé aplikace (např. Blender) také umožňují zadávat souřadnice objektu přímo, což lze považovat za užitečné, potřebujeme-li objekty umístit přesně. Ráda bych tedy umožnila uživateli obě možnosti – drag and drop a přímé zadávání souřadnic.

Příhodná je také možnost pohybovat objekty po mřížce (tj. souřadnice objektu mohou být např. pouze násobky deseti), abychom mohli lépe objekty umístit např. do jedné přímky.

3.3 Posun scény

Po vložení několika objektů na scénu potřebujeme občas posunout celou scénou. To je v aplikaci SketchUp vyřešeno tlačítkem s obrázkem ruky (viz obr. 3.2), po jehož vybrání můžeme pohybovat celou scénou táhnutím myši v požadovaném směru.

Tento způsob obsluhy posunu scény je dle mého názoru nevhodný, nebot' musíme přepnout od aktuálně provozované činnosti (např. přidávání objektu),



Obrázek 3.2: Lišta s funkcemi aplikace SketchUp (vybrán posun scény).

poté posunout scénu a přepnout se znovu na předešlou činnost. Praktičtější řešení poskytuje aplikace ROS, kde scénu posouváme táhnutím myši se stisknutým pravým tlačítkem, nebo původní verze, kde se posunuje táhnutím myši s držením klávesy `shift`.

3.4 Promítání

Aplikace, sloužící k modelování ve 3D prostoru, typicky využívají jednobodového perspektivního promítání (ale např. Blender poskytuje možnost přepnout do ortogonálního promítání), což je vhodné pokud chceme zachovat informaci o vzdálenosti objektu od pozorovatele a nevadí nám zanedbání rovnoběžnosti. Pro simulování optických jevů a sledování chování paprsků se však jeví jako vhodnější promítání ortogonální, kde je naopak zanedbána vzdálenost objektu od pozorovatele, ale rovnoběžnost zůstává.

3.5 Přibližování

Přibližování scény je obvykle zajištěno kolečkem myši (občas v kombinaci s klávesou `shift`, nebo `ctrl`), což soudím jako lepší způsob, než mít v menu umístěné tlačítko + pro přibližování a − pro oddalování.

V Blenderu se přibližuje vždy do středu obrazovky (tzn. střed při libovolné změně měřítka zůstane na stejném místě), kdežto v ostatních zkoumaných programech se přibližuje tam, kde je aktuálně umístěný kurzor myši. Druhý způsob mi přijde uživatelsky přívětivější – lze si jednodušeji vybrat, která část pracovní plochy bude přiblížena.

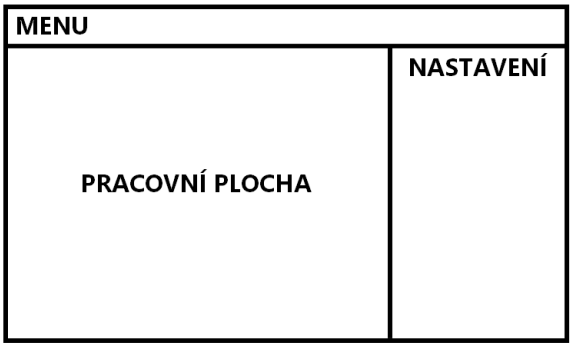
4 Návrh GUI

4.1 Rozložení okna

Nejdůležitějším požadavkem na rozložení okna je přehlednost – veškeré akce a nastavení, především ty často používané, musí být dobře dostupné, aby je uživatel nemusel hledat. Tomuto požadavku vyhovuje původní verze, a proto bude rozložení okna ponecháno – v levé části pracovní plocha, v pravé panel nastavení (tento panel by měl obsahovat veškerá možná nastavení pro konkrétní objekt). Dále bude mít aplikace horní menu, ze kterého bude přístupné globální nastavení, které nesouvisí s konkrétními objekty.

V původní verzi lze pracovní plochu libovolně rozdělovat, chceme-li pracovat na více stolech najednou, avšak při rozdělení na větší počet stolů je prostor pro jeden stůl značně omezený. Při zkoumání aplikací zabývajících se 3D modelováním jsem nabyla dojmu, že 3D scéna potřebuje více prostoru pro přehlednější manipulaci s objekty, než kolik poskytuje způsob dělení plochy v původní 2D verzi.

Možnost pracovat na více stolech najednou je ovšem nutné zachovat, neboť pracovat s jediným stolem by bylo značné omezení funkcionality v porovnání s původní verzí. Tento problém je možné vyřešit například použitím záložek (jak známe z webových prohlížečů). Starý způsob dělení plochy na více stolů hodlám také ponechat – pracujeme-li totiž s menším počtem objektů na jednotlivých stolech, není nutné, aby měl každý stůl vlastní záložku. Jednoduché schéma navrhovaného okna můžeme vidět na obrázku 4.1



Obrázek 4.1: Návrh rozložení okna.

4.2 Dostupné úkony

V této kapitole si ujasníme, jaké všechny úkony budou v aplikaci uživateli umožněny.

Úkony nad objektem

Základním prostředkem k simulaci paprskové optiky je modelování 3D scény a proto musí aplikace nabízet přidávání (resp. mazání) optických objektů na scénu, změnu jejich velikosti, pozice a natočení. Poslední dva úkony by bylo vhodné umožnit jednak akcemi myši (kvůli rychlosti a přístupnosti práce se simulátorem), tak i nastavením konkrétní pozice a úhlu natočení v postranním panelu, pokud by uživatel chtěl přesně nastavit vzájemné pozice několika objektů v optické soustavě.

V aktuální verzi jádra [3] jsou dostupné následující objekty: zrcátko, světlo (bodové a plošné), holografická mřížka a stínítko. Bylo by vhodné implementovat GUI tak, aby bylo jednoduché do něj přidat další optický objekt v případě, že by byl přidán do jádra. Jak bylo řečeno v předchozím odstavci, každému objektu lze nastavit pozici a natočení. Ovšem některé objekty mají speciální vlastnosti, se kterými je třeba v GUI počítat:

- *zrcátko, stínítko, mřížka* – velikost objektu,
- *mřížka* – rozlišení ([3], str. 42), volba difrakčních maxim, tlačítko pro vytvoření záznamu,
- *světlo* – přepínání mezi bodovým a plošným světlem, přepínání mezi monochromatickým a polychromatickým a odpovídající nastavování vlnových délek, hustota paprsků,
- *světlo bodové* – vrcholový úhel,
- *světlo plošné* – velikost svazku (v jádře je svazek reprezentován čtvercem, velikost svazku je tedy velikost strany tohoto čtverce).

Představuji si, že by bylo vhodné umožnit uživateli uzamknout pohyb objektu ve směru některé osy KSS.

Úkony nad scénou

Kromě manipulace s objekty budeme potřebovat i prostředky k manipulaci s pracovní plochou (scénou). Určitě je nezbytné poskytnout uživateli možnost scénu posunout a rotovat s ní okolo os KSS. Dále by mohla být užitečná změna měřítka, tedy přibližování a oddalování scény. Představuji si, že tyto akce bude možné provádět jednoduchými zkratkami na klávesnici nebo myši, protože je pravděpodobné, že je uživatel bude potřebovat často a bylo by tedy vhodné, aby byly lehko přístupné.

V předchozí kapitole (viz 4.1) jsem zmiňovala, že hodlám zachovat možnost dělit pracovní plochu. Předpokládám, že příkazy na dělení plochy budou v jejím kontextovém menu (vyvolání kliknutím pravého tlačítka myši). V tomto kontextovém menu mohou být i další příkazy, např. přidávání a mazání objektů, nebo jejich kopírování a vkládání.

Úkony nad aplikací

Úkony nad aplikací jsou ty, které jsou přístupné z hlavního menu aplikace. Především se jedná o vytváření nových stolů a záložek (aby mohl uživatel pracovat na více simulacích zároveň), dále pak různá globální nastavení aplikace:

- plánuji ponechat jazykovou lokalizaci z původní verze,
- přepínání mezi zobrazením/skrytím paprsků, které neprotínají žádný objekt,
- změna barev objektů ,
- nastavení rychlosti rotace,
- průhlednost paprsků.

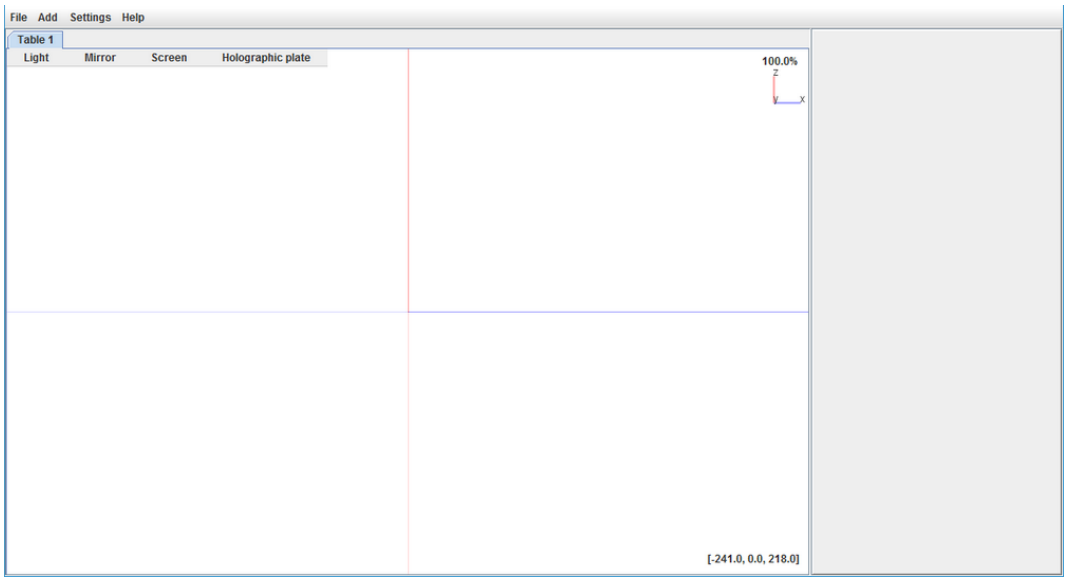
Vhodnou funkcionalitou by také bylo uložení scény do souboru (a načítání ze souboru), případně exportování scény ve formátu `svg`.

5 Implementace

V této kapitole bude nejprve představena implementovaná aplikace vytvořením a rekonstrukcí hologramu (kap. 5.1) – obdobným způsobem jako byla představena původní aplikace v kapitole 2.2.2. Následně bude popsáno, jak byla aplikace implementována (kap. 5.2), tj. jaké způsoby (a proč) implementace byly zvoleny a některé zajímavé implementační detaily.

5.1 Představení aplikace

Po spuštění se zobrazí okno aplikace – vidíme na obrázku 5.1 – rozložení okna odpovídá návrhu z kap. 4.1, není třeba se jím znovu zabývat. Dělá-li někomu výchozí anglický jazyk problémy, lze GUI přepnout do českého jazyka.

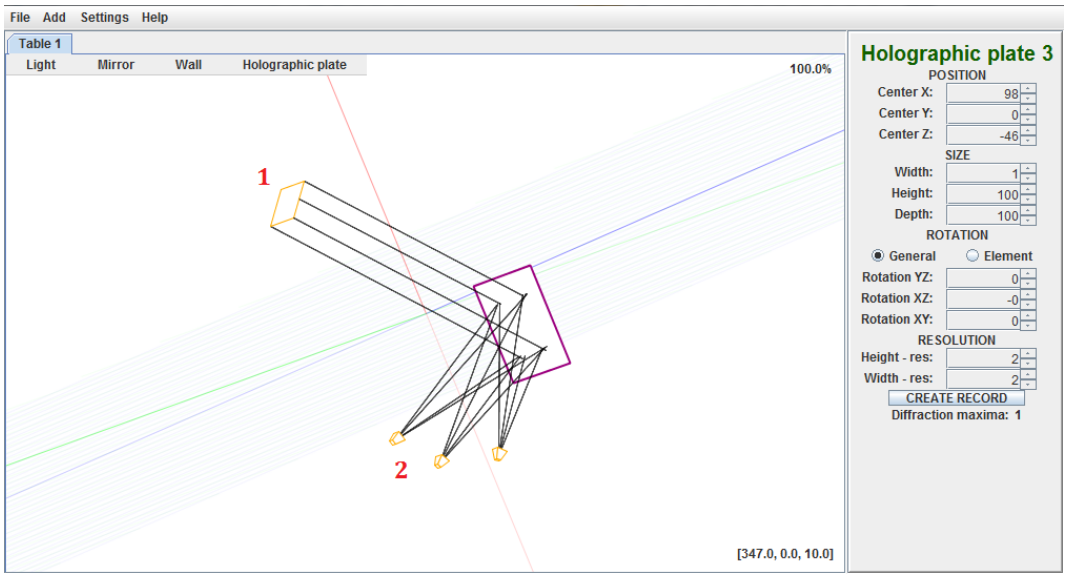


Obrázek 5.1: Okno aplikace po spuštění.

Představíme si nyní GUI aplikace při tvorbě a rekonstrukci hologramu. Jak přesně se která akce provede, bude vysvětleno později v uživatelské příručce. V této kapitole jde o to, aby bylo vidět, jak GUI vypadá a jaké možnosti nabízí. Např. si zde neukážeme, jak se rotuje pohledem kolem os KSS, či jak je tato funkce implementována, pouze si povíme, že rozhraní tuto funkci nabízí.

5.1.1 Vytvoření záznamu hologramu

Nejprve je nutné přidat na stůl potřebné optické objekty – světlo, které bude sloužit jako referenční **(1)**; světlo (či svazek několika světel), které zastupuje objekt, jehož záznam je vytvářen **(2)**; záznamový materiál, tj. holografickou desku – například tak, jak je vidět na obrázku 5.2. Dále aplikace nabízí stínítko (zeď) a zrcadlo, ale tyto objekty nyní nejsou potřeba. Nevyhovují-li uživateli výchozí barvy vykreslených optických objektů, lze je změnit v nastavení.



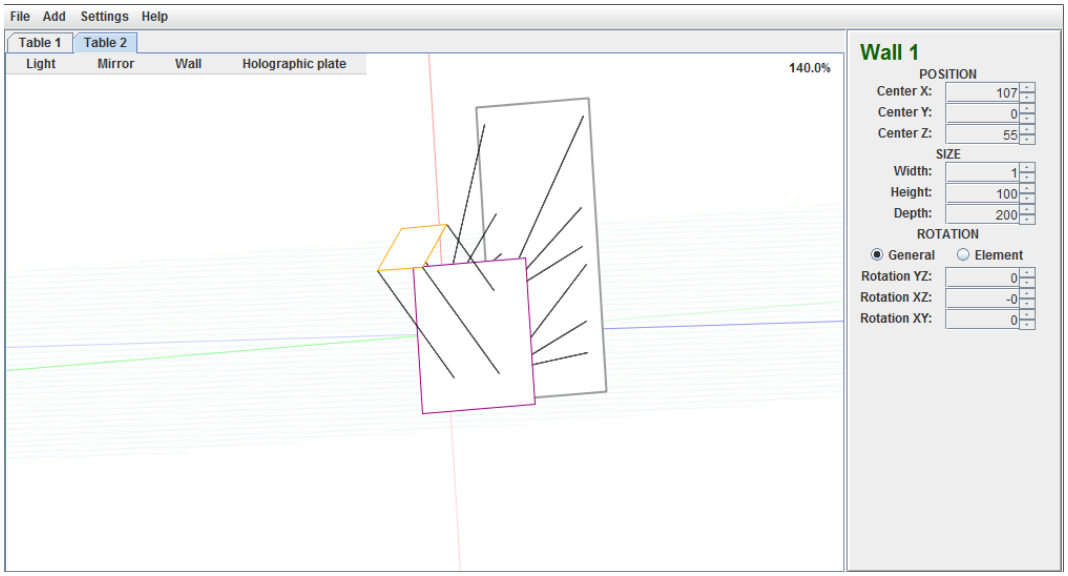
Obrázek 5.2: Objekty nutné pro záznam hologramu.

Pohled na stůl a pozice a natočení objektů nejsou ve výchozí poloze, neboť v aplikaci je možné pohybovat a rotovat okolo os KSS jak pohledem, tak jednotlivými objekty (kterým lze také měnit velikost). Dále je možné měnit přiblížení pohledu, nebo jej lze klávesovými zkratkami okamžitě přepnout do roviny `XY`, `XZ`, nebo `YZ`. V nastavení taktéž lze změnit krok (ve stupních), o který budou objekty (či pohled) rotovat jedním stiskem klávesové zkratky.

Po přidání objektů na stůl vytvoříme záznam hologramu (aplikace při vytváření záznamu vyzve k výběru referenčního světla, tzn. zde **(1)**) a můžeme přejít k rekonstrukci.

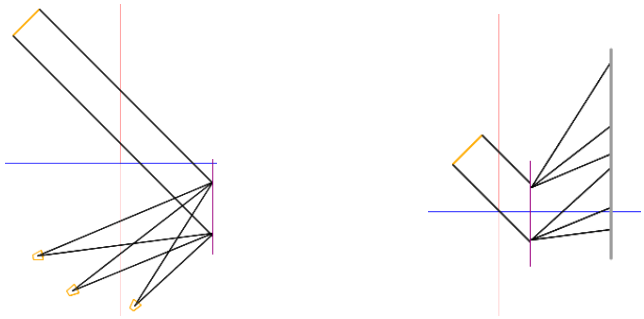
5.1.2 Rekonstrukce hologramu

Vytvoříme nový stůl (stejně jako v původní aplikaci, i zde je možné vytvářet stoly a pohledy na ně – ovšem nová aplikace navíc umí kromě dělení plochy taktéž vytvářet záložky), na který zkopírujeme holografickou desku a referenční světlo. Za desku umístíme stínítko (`Wall`), aby se měly paprsky, vycházející z holografické desky, kam promítat. Výsledek vidíme na obr. 5.3.



Obrázek 5.3: Rekonstruovaný hologram.

Pro názornost na obr. 5.4 vidíme vlevo bokorys (resp. pohled z roviny `XZ`) soustavy před záznamem hologramu a poté vpravo rekonstruovaný hologram.



Obrázek 5.4: Bokorys optické soustavy pro záznam hologramu (vlevo) a rekonstruovaného hologram (vpravo).

5.2 Vlastní implementace

5.2.1 Získání dat

Před samotným vykreslením požadované scény je nutné získat z výpočetního jádra veškeré potřebné údaje o objektech k vykreslení. Každý stůl (třída `Box` představující jednu konkrétní simulaci) si uchovává několik seznamů, které jsou důležité pro výpočet paprsků, z čehož pro zobrazení jsou klíčové právě dva seznamy – seznam všech již vypočtených paprsků (třída `Ray`) a seznam všech ostatních objektů (třída `Element`[1]) mezi které patří např. světlo, nebo zrcátko. Každá instance panelu pro zobrazení (třída `View`) si proto uchovává příslušný `Box`, kde má dostupné oba seznamy.

5.2.2 Předzpracování

Každý objekt je reprezentován sítí vrcholů, které jsou vypočtené ze souřadnic středu a rozměrů objektu. Tyto body jsou poté uspořádány do několika množin, z nichž každá odpovídá jedné stěně objektu (například objekt tvaru kvádru má 8 vrcholů, ze kterých je vytvořeno 6 stěn).

Prvním krokem před vykreslením je tvorba seznamu segmentů – z každé stěny každého objektu je vytvořen segment, který odpovídá jedné stěně objektu. Jeho souřadnice bodů jsou případně transformovány, pokud je objekt, kterému segment (resp. stěna) náleží, natočen[2]. Na segmenty jsou také přepočítány osy KSS a veškeré paprsky.

5.2.3 Viditelnost

Pořadí vykreslení segmentů je určeno strukturou zvanou BSP strom (z anglického *Binary Space Partition tree* [4]), která je před vykreslením vytvářena ze seznamu segmentů následujícím rekurzivním postupem:

[1]Pojmem `Element` je myšleno souhrnné označení pro všechny optické objekty (vyjma paprsků). V implementaci je ale k těmto objektům přistupováno pomocí abstraktní třídy `ElementController`.

[2]Pro minimalizaci zaokrouhlovacích chyb jsou všem objektům uchovány pouze souřadnice jeho středu, tvar, rozměry a transformační matice a až při vykreslení jsou tyto údaje přepočteny na konkrétní souřadnice vykreslované stěny.

1. Vybereme náhodný segment ze seznamu, který bude kořenem aktuálního podstromu (první takto vybraný segment je tedy kořenem celého stromu).

2. Vypočteme obecnou rovnici roviny pomocí vektorového součinu, ve které se segment nachází.

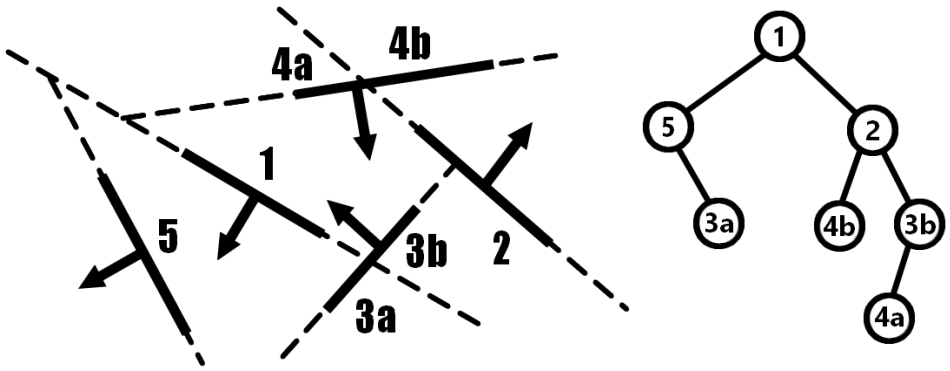
3. Pro každý další segment ze seznamu určíme, na které straně této roviny leží – takto tedy vytvoříme dva nové seznamy, z nichž jeden umístíme jako levého potomka kořenu, druhý jako pravého potomka[3]. Pokud se stane, že rovina nějakým segmentem prochází, bude tento rozdělen na dva segmenty.

4. Pokračujeme bodem 1 pro seznam segmentů v levém potomkovi, pokud má více než 1 prvek. To samé provedeme pro seznam v pravém potomkovi.

Příklad sestavení stromu pro konkrétní scénu můžeme vidět na obrázku 5.5. Je nutné podotknout, že z podstaty algoritmu se nejedná o ideální rozložení stromu (segmenty nejsou předem nijak analyzovány a řazeny) a v tomto konkrétním případě je možné se zcela vyhnout dělení segmentů na části. Nicméně zde máme velmi zjednodušený příklad (na obr. 5.5 jsou všechny segmenty kolmé na půdorysnu), v reálné scéně může být segmentů větší počet a mohou být vůči sobě různě natočeny. Analyzování scény za účelem výběru vhodného kořenu aktuálního podstromu by proto bylo složité a algoritmus by se tím zbytečně zkomplikoval. Dosáhli bychom sice menšího počtu řezů, ale pravděpodobně za cenu zvýšení časové a paměťové náročnosti algoritmu.

Tento algoritmus je dostačující pro segmenty (tj. dvojrozměrné objekty určené alespoň třemi body, které neleží v jedné přímce), z nichž každý leží právě v jedné rovině – z popisu algoritmu je zřejmé, že je nezbytně nutné, aby tato rovina byla pro každý segment jednoznačně určitelná (protože v kroku 3 musíme určit, jestli ostatní segmenty leží před nebo za rovinou, určenou segmentem, který je právě zpracováván).

[3]To, jestli má být segment přidán do levého (resp. pravého) seznamu, určíme dosazením bodů segmentu do obecné rovnice roviny segmentu původního a pokud vyjde kladné číslo, pak je segment poslán do levého potomka, jinak do pravého potomka. Dodržování této konvence je později nutné pro správné vykreslování.



Obrázek 5.5: Několik segmentů v pohledu shora (vlevo) a odpovídající BSP strom (vpravo), převzato z [4].

Z toho ovšem vyplývá, že algoritmus není v této podobě použitelný pro výpočet viditelnosti paprsků a os (tj. jednorozměrné objekty, dále přímky[4]), poněvadž tyto objekty leží v nekonečně mnoha rovinách a tedy nemůžeme jednoznačně určit rovinu, ve které leží – s přímkami nelze nakládat jako s dvourozměrnými objekty (segmenty).

Pochopitelně ale potřebujeme, aby byla viditelnost vypočtena i pro jednorozměrné objekty – proto byla zavedena úprava struktury stromu – každý uzel stromu má kromě levého a pravého potomka typu *Segment* navíc ještě levého a pravého potomka typu *seznam přímek* – přímky z jednoho seznamu jsou vykresleny před objektem z rodičovského uzlu, z druhého seznamu za ním.

Tuto úpravu struktury si můžeme dovolit – v případě jednorozměrných objektů je totiž možné, aby byl v jediném uzlu stromu seznam objektů (nikoliv pouze jediný objekt), protože přímky můžeme vykreslit v libovolném pořadí – co se viditelnosti týká, nijak se vzájemně neovlivňují a mohou se jakkoliv protínat. Důležité je pouze určit, v jakém seznamu se každá přímka nachází – resp. před a za jakými objekty bude vykreslována. Každopádně stejně jako u dvojrozměrných segmentů, i zde se může stát, že přímka rovinu (určenou nějakým segmentem) protíná – v takovém případě bude rozdělena na dvě (a každá část bude v jiném seznamu).

Výše popsané kroky algoritmu výpočtu viditelnosti dvourozměrných objektů není třeba nijak upravovat. Výpočet viditelnosti pro jednorozměrné objekty probíhá na již hotovém stromu viditelnosti dvourozměrných segmentů

[4]Přímky ale nejsou přesné označení, protože ve většině případů jde o polopřímky, nebo pouze úsečky

– přímky pouze přidáváme na příslušná místa do stromu, což probíhá následovně:

1. Na vstupu máme:
   - seznam všech přímek (v simulátoru jsou všechny paprsky polopřímky vycházející z určitého objektu do prostoru (resp. úsečky vedoucí do dalšího objektu); osy KSS jsou polopřímky z bodu [0, 0, 0]),
   - hotový BSP strom dvourozměrných objektů (výpočet začíná u kořene stromu).
2. Pro každou přímku z aktuálního seznamu zjistíme vzájemnou polohu s aktuálním uzlem stromu, přičemž mohou nastat podobné situace jako u viditelnosti samotných segmentů:
   - přímka leží celá na jedné straně roviny, kterou určuje aktuální segment, a není ji třeba dělit (je s rovinou rovnoběžná),
   - přímka rovinu segmentu protíná a je ji nutné rozdělit v bodě, který leží v této rovině.
3. Seznam přímek rozdělíme na dva seznamy tak, aby všechny přímky v jednom seznamu ležely na téže straně roviny. Těm, které leží před rovinou (po dosazení jakéhokoli bodu přímky do obecné rovnice roviny je výsledkem kladné číslo), budeme říkat levé přímky (protože se budou dále týkat pouze levého potomka), těm za rovinou pravé přímky.
4. Pokud aktuální prvek již levého, resp. pravého potomka nemá, jsou mu tyto přímky nastaveny jako jeho levé, resp. pravé přímky a algoritmus pro tento prvek končí.
5. V opačném případě pokračujeme bodem 2 pro levého potomka a levé přímky a následně pravého potomka a pravé přímky.

Po skončení algoritmu jsou tedy všechny přímky přiřazeny listům původního stromu buď jako levé přímky (mají být vykresleny před tím, než vykreslíme daný segment), nebo jako pravé přímky (jsou vykresleny ihned po vykreslení daného segmentu).

5.2.4 Vykreslení

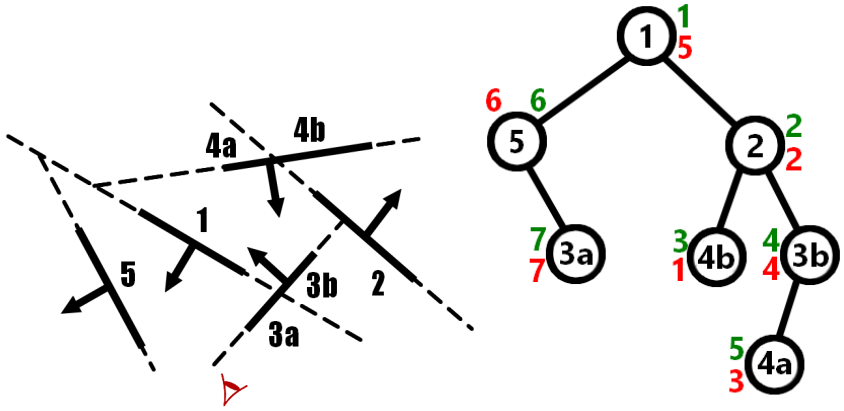
Vykreslení jednotlivých segmentů probíhá podle pořadí určeného správným procházením stromu viditelnosti. Klíčovým krokem je správný výběr bodu, ze kterého chceme pozorovat vykreslenou scénu. Jelikož je výchozím natočením pohled na rovinu XZ a kladná část osy y směřuje před obrazovku, je jako pozorovací bod zvolen bod o souřadnicích [0, MAX, 0], kde `MAX` je největší souřadnice, kam lze umístit objekt (tím je zaručeno, že žádný objekt nebude ležet za pozorovacím bodem). Důležité je také podotknout, že BSP strom se přepočítává pouze při změnách na objektech (posun, změna velikosti, otočení) a s natáčením pohledu na scénu se tedy nijak nemění. Proto je nutné pro procházení stromem pozorovací bod otočit do roviny pohledu (o rotacích a posunech bodů dále v kapitolách **Rotace** a **Posun**).

Algoritmus procházení stromu pak probíhá prohledáváním do hloubky (začínáme kořenem stromu) podle následujících pravidel:

1. Dosadíme (otočený) pozorovací bod do obecné rovnice roviny aktuálního uzlu (segmentu), dále nás zajímá pouze znaménko výsledku.
2. Pokud byl výsledek kladný, pokračujeme prohledáváním pravého podstromu bodem 1; jinak pokračujeme taktéž bodem 1, ale pro levý podstrom[5]. Pokud aktuální uzel již nemá pravého (resp. levého) potomka, ověříme, zda-li má pravé (resp. levé) přímky a pokud ano, vykreslíme je. Následně vykreslíme i segment v aktuálním uzlu a pokračujeme bodem 3.
3. Nyní prohledáváme ten podstrom, který jsme neprohledávali v druhém kroku. Tedy pokud byl výsledek kladný, pokračujeme prohledáváním levého podstromu bodem 1, jinak pokračujeme pravým podstromem. Pokud aktuální uzel již nemá levého (resp. pravého) potomka, ověříme, zda-li má levé (resp. pravé) přímky a pokud ano, vykreslíme je.

Konkrétní pořadí prohledávání a vykreslování stromu můžeme vidět níže na obrázku 5.6. Zelená čísla představují pořadí prohledávání uzlů stromu, červená čísla pořadí vykreslení.

[5]Jinými slovy: vždy chceme začít prohledáváním toho poloprostoru, ve kterém pozorovací bod neleží, neboť vykreslujeme na základě principu malířova algoritmu (nejdříve se vykreslují segmenty dále od pozorovatele).



Obrázek 5.6: Ukázka procházení stromu viditelnosti (vpravo) pro konkrétní pozorovací bod scény (vlevo).

Rotace

Pro transformace z výchozích bodů do bodů po konkrétním natočení si zavedeme tzv. *transformační matice* [4]. Jedná se o matice pracující s homogenními souřadnicemi, které obecně pomáhají při přepočtu skutečných bodů modelu na body, které vykreslujeme. Pro třírozměrný prostor mají velikost 4×4, kde čtvrtý řádek, resp. sloupec představuje právě homogenní souřadnice. Samotné rotace docílíme maticemi na vzorcích 5.1, 5.2 a 5.3, kde každá provádí rotaci právě okolo jedné z os.

$$R_x(\alpha) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\alpha) & \sin(\alpha) & 0 \\ 0 & -\sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{5.1}$$

$$R_y(\alpha) = \begin{pmatrix} \cos(\alpha) & 0 & -\sin(\alpha) & 0 \\ 0 & 1 & 0 & 0 \\ \sin(\alpha) & 0 & \cos(\alpha) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{5.2}$$

$$R_z(\alpha) = \begin{pmatrix} \cos(\alpha) & \sin(\alpha) & 0 & 0 \\ -\sin(\alpha) & \cos(\alpha) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{5.3}$$

Posun

Pro otočení objektu okolo jeho středu tyto matice nestačí, neboť rotují bodem okolo středu souřadného systému. Zavedeme si proto ještě matici posunu (viz vzorec 5.4), kterým se posuneme do středu objektu (vektor v odpovídá souřadnicím středu objektu).

$$P(v) = \begin{pmatrix} 1 & 0 & 0 & v_x \\ 1 & 0 & 0 & v_y \\ 1 & 0 & 0 & v_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{5.4}$$

Další nezbytnou operací jsou inverzní transformace. Pro rotaci jsou tyto matice shodné s maticemi původními po transponování (viz vzorec 5.5), u posunu pak místo vektoru v použijeme vektor opačný (viz vzorec 5.6).

$$R^{-1}(\alpha) = R^T(\alpha) \tag{5.5}$$

$$P^{-1}(v) = P(-v) \tag{5.6}$$

Celkovou transformací je tedy posun do středu objektu, požadovaná rotace a posun zpět do středu souřadného systému. K tomu využijeme skládání matic, které provedeme postupným násobením jednotlivých matic a výsledná transformační matice T vypadá takto:

$$T = P(v) * R(\alpha) * P^{-1}(v) \tag{5.7}$$

Promítání

Aplikace využívá ortogonální promítání, což v praxi znamená, že vykreslujeme pouze dvě souřadnice a třetí je zanedbána.

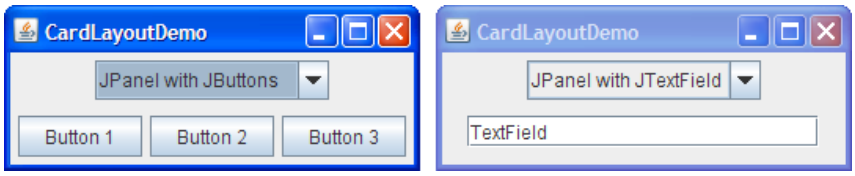
5.2.5 Rozložení okna

Rozložení okna bylo zachováno z původní verze, tedy nahoře se nachází menu, vpravo je místo pro panel s nastavením a největší část plochy vlevo zaujímá pracovní plocha pro vykreslování optických modelů. Novinkou je způsob správy několika paralelních stolů, resp. pohledů. V původní aplikaci musel uživatel uspořádat veškeré aktuálně potřebné pohledy na jedinou pracovní plochu. V nynější aplikaci je možné vytvářet záložky (podobně jako ve webových prohlížečích) – tím uživatel získá k dispozici více pracovních ploch najednou.

Záložky jsou implementovány pomocí komponenty `JTabbedPane` (viz obr. 5.7).

Obrázek 5.7: Ukázka použití JTabbedPane [6].

Možnou alternativou této komponenty bylo využití tzv. `CardLayout`. Ten ovšem vyžaduje použití další komponenty – tzv. *přepínače*, kterým se přepíná zobrazení jednotlivých *karet* (`CardLayout` si lze představit jako balíček karet, z nichž vidíme vždy pouze tu jednu, která je na vrchu). Na obr. 5.8 vidíme ukázku implementovaného `CardLayout`, kde jako přepínač byl zvolen `JComboBox` (rozbalovací nabídka, kde každá položka odpovídá jedné *kartě*).

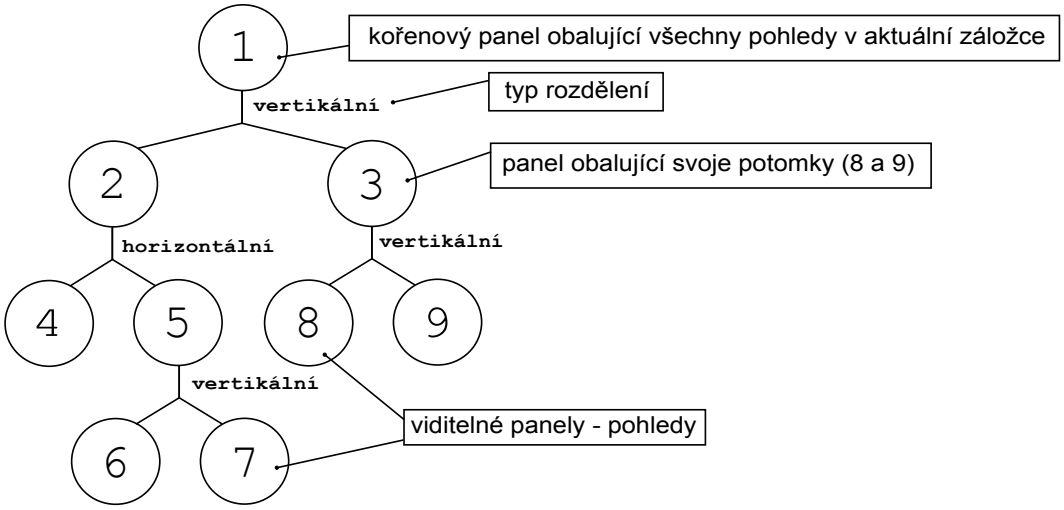


Obrázek 5.8: CardLayout s JComboBox přepínačem [6].

V komponentě `JTabbedPane` není třeba přepínač implementovat – implementace je jednodušší pro programátora. Dále přepínání mezi jednotlivými záložkami probíhá pouze jedním klikem na požadovanou záložku – `JTabbedPane` je tedy vhodnější i z hlediska uživatelského.

Z návrhu dále vyplynulo, že by bylo vhodné ponechat i dělení pracovní plochy na několik pohledů, čehož jsem dosáhla komponentou `JSplitPane`. Ta umožňuje vložit dvě jiné komponenty vedle sebe (resp. jednu nad druhou) do jednoho panelu[6] a poskytuje uživateli dělič (`JSplitPane divider`), kterým má možnost měnit velikost jedné komponenty vůči druhé v rámci obalovacího panelu.

Pro lepší správu těchto rozdělení (tzn. přidávání a mazání) jsou jednotlivé panely uloženy ve struktuře znázorněné na obr. 5.9 připomínající binární strom. Kořenový prvek (1) představuje panel, který obaluje všechny pohledové panely. Pohledovým panelem se rozumí panel, do kterého se již vykresluje a dále se nedělí. Typ rozdělení určuje, jakým způsobem se umístí do daného panelu jeho potomci, tedy horizontální umisťuje prvního potomka nahoru a druhého dolu, ve vertikálním je první vlevo a druhý vpravo.



Obrázek 5.9: Struktura pro správu rozdělování pracovní plochy.

Aby bylo možné libovolně rozdělovat pohledy, měnit jim velikosti a poté je mazat, musí po celou dobu tato struktura splňovat několik vlastností:

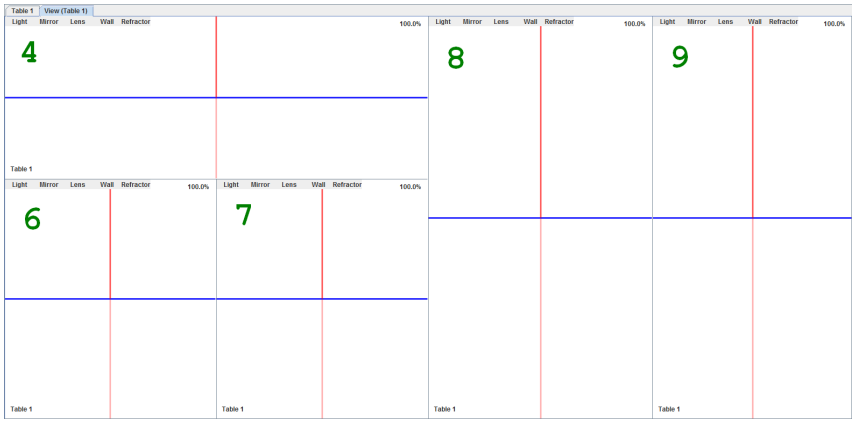
- pohled (který není kořenovým panelem) má právě jednoho rodiče,
- panel má právě dva, nebo žádného potomka (jeden potomek není dovolen,

[6]Pojmem `panel` je zde i jinde v textu obecně myšlena komponenta dědící komponentu `Container`, která obaluje další komponenty a přitom se sama nijak viditelně nezobrazuje uživateli.

neboť by se jednalo o redundantní panel - v rodičovském panelu by byl jeho potomek přes celou jeho velikost),

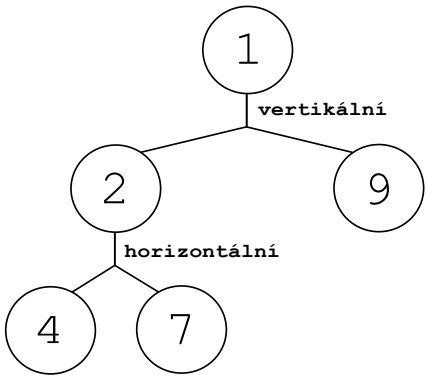
- panel, který nemá žádného potomka je pohledem (uživateli viditelný panel, ve kterém již lze modelovat optické soustavy).

Pomocí výše zmíněného diagramu lze docílit stejného rozdělení jako je na obrázku 5.10. Výchozí (kořenový) pohled rozdělíme vertikálně, čímž získáme nové pohledy: levý a pravý. Následně se rozdělí pravý pohled vertikálně, levý pohled horizontálně a z tohoto horizontálního rozdělení ještě rozdělíme spodní panel vertikálně. Nyní jsou uživateli zobrazeny pouze pohledové panely 4, 6, 7, 8 a 9, neboť ostatní panely (1, 2, 3 a 5) slouží pouze jako pomocné (obalovací) z důvodu zachování struktury například při změně velikostí, nebo mazání.

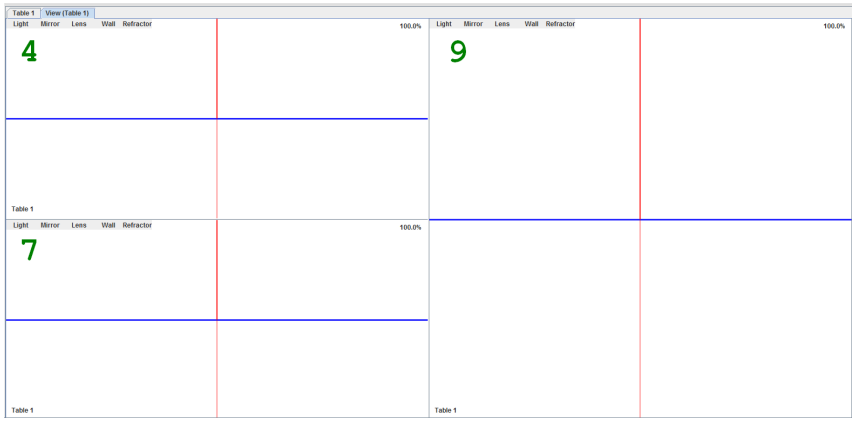


Obrázek 5.10: Rozdělení pracovní plochy podle diagramu na obrázku 5.9

Při mazání pohledů je důležité obnovit vlastnosti struktury. Smazáním například panelu 6 bychom se dostali do situace, kde panel 5 má pouze jednoho potomka (7). V tomto momentě je tedy panel 5 redundantní a nahradí se jeho jediným potomkem, panelem 7. Pokud bychom smazali ještě například panel 8 (a aplikovali stejný postup obnovy vlastností), získáme diagram z obrázku 5.11 a jemu odpovídající realizaci na obrázku 5.12.



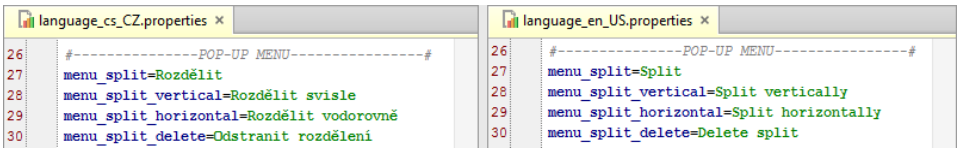
Obrázek 5.11: Struktura po smazání panelů 6 a 8.



Obrázek 5.12: Rozdělení pracovní plochy podle diagramu na obrázku 5.11

5.2.6 Jazyková lokalizace

Z původní verze byla dále zachována jazyková lokalizace. Třída `Language`, která dědí `Observer`, zajišťuje získávání všech popisků, které lze měnit implementací rozhraní `Observable` za běhu aplikace. Samotné lokalizace pak dosáhneme pomocí `ResourceBundle`, ze kterého se texty získávají na základě klíčového hesla, které je pro daný popisek ve všech lokalizacích stejné. Jak vypadají soubory můžeme vidět na obrázku 5.13.



Obrázek 5.13: Soubory s jazykovou lokalizací (vlevo česky, vpravo anglicky).

Metoda `getText`, dodávající požadovaný text, přijímá dva parametry:

- klíč podle kterého chceme najít popisek,
- alternativní popisek, který chceme použít v případě neúspěchu nalezení podle klíče.

Alternativním popiskem jsem zamezila pádu programu externím zásahem – například smazáním souborů s lokalizací, nebo překlepem v klíči.

5.3 Rozšíření

Aplikace byla navržena tak, aby grafické uživatelské rozhraní i výpočtové jádro bylo snadno rozšiřitelné o další optické objekty. Ukážeme si, jak by se simulátor rozšířil například o čočku; budeme předpokládat, že v jádře je již čočka implementována (pro detailní popis rozšiřitelnosti výpočetního jádra viz [3]).

Nejprve je nutné umožnit přidání čočky na scénu, tedy přidání položky do kontextového menu a do menu rychlého přístupu (levý horní roh každého pohledu). Pro přidání čočky do kontextového menu vytvoříme třídu `AddLens` v balíku `gui.elementAction`, která dědí třídu `AddElement`. V této třídě implementujeme metodu `createElement`, která požádá rozhraní `LensController` o vytvoření čočky s výchozími hodnotami (podobně jako je tomu například ve třídě `AddLight` pro objekt typu *světlo*), a nastavíme požadované popisky a klávesové zkratky. Nyní již můžeme přistoupit k samotnému přidání tlačítek do `gui`. Ve třídě `PopupMenu` (v balíku `gui.menu`) vytvoříme komponentu typu `JMenuItem` a přidáme jí komponentě `addElement`. Poté vytvoříme komponentu `lensButton` typu `ToggleButton` ve třídě `View` (v balíku `gui.workspace`) a v metodě `addToolBar` ji přidáme stejným způsobem jako ostatní komponenty. Nakonec upravíme `ActionListener` ve třídě `ToggleButton`, který zajistí správnou funkcionalitu výběru tlačítka, a ve třídě `ViewLis-`

`tener` (balík `gui.workspace`) v metodě `mousePressed` dopíšeme *else-if* větev pro přidání objektu na scénu.

Dalším krokem je implementace postranního panelu pro daný objekt. To provedeme implementací třídy `LensPanel`, která dědí třídu `ElementPanel` (balík `gui.sidebar`). Výchozími prvky panelu jsou tlačítka pro nastavení pozice objektu na scéně a tlačítka pro rotaci objektu, které jsou již implementovány ve třídě `ElementPanel`. Při rozšiřování simulátoru o čočku je tedy nutné implementovat jen tlačítka, která jsou specifická pro tento typ optického členu. Poté je nutné dopsat *else-if* větev do třídy `InfoPanel` (balík `gui.sidebar`), která na základě typu objektu zobrazuje daný typ postranního panelu.

Posledním krokem rozšiřování simulátoru je dopsání veškerých popisků, které se mají aktualizovat při změně jazyka. Ve všech potřebných výše zmíněných třídách jsou připraveny metody *update*, kam dopíšeme požadovaným tlačítkům nový popisek příkazem ve tvaru *lensButton.setText(getText("menu_add_lens", "Lens"));* (nalezení hodnoty, která náleží klíči „menu_add_lens“, v souboru s vybraným jazykem a její nastavení tlačítku `lensButton`; slovo „Lens“ se použije, pokud nebude v souboru požadovaný klíč nalezen).

Co se týká samotného vykreslování, není nutné simulátor nijak upravovat. Pro vykreslovací algoritmy není typ objektu důležitý, protože jejich vstupem je pouze seznam plošek a čar.

6 Testování

Aplikace byla testována na skupině 6 uživatelů méně, či více seznámených s danou problematikou, kde každý tester byl požádán o namodelování jednoduché soustavy a následné vytvoření a rekonstrukci hologramu dle dodaného návodu (viz příloha B). Návod je strukturovaný do několika bodů, které přibližně odpovídají bodům návrhu aplikace, především bodů týkajících se ovladatelnosti grafického uživatelského rozhraní. Rozsahem i stylem míří k velmi zestručněné uživatelské příručce, což otestovalo intuitivní ovládání výsledného programu.

Všem testerům přišla aplikace uživatelský přívětivá a intuitivní na ovládání. Na žádné veliké chyby, které by znemožňovaly v nějaké míře s aplikací pracovat, nenarazili[1]. Objevili pouze pár drobných nedostatků, které byly ovšem následně opraveny. Nyní budou popsány tyto nedostatky a jak byly vyřešeny.

První nedostatek se týká menu rychlého přístupu (menu v levém horním rohu každého pohledu, pomocí kterého uživatel může přidávat optické objekty na plochu). Tlačítka byly implementovány jako `ToggleButton`, tedy tlačítka, která jsou po prvním kliknutí aktivovaná a pro deaktivaci je nutné na ně kliknout znovu. Uživatel méně seznámený s aplikací tedy vybral např. světlo, vložil jej kliknutím na plochu a dále pokračoval v práci (například nastavil vlnovou délku), ale již nedeaktivoval vybrané tlačítko. Každým kliknutím do plochy si pak omylem přidal nové světlo, dokud toto tlačítko nedeaktivoval. V aktuální verzi simulátoru má tedy toto tlačítko dvě funkce: pouhým kliknutím na požadovaný typ objektu uživatel vloží tento objekt právě jednou a tlačítko se následovně deaktivuje samo; pokud toto tlačítko ale uživatel vybere za stisknutí klávesy SHIFT, tlačítko se zamkne a uživatel opět vkládá objekty tak dlouho, dokud jej sám nedeaktivuje opětovným kliknutím.

Dalším nedostatkem byl problém místy poznat, na co vše lze kliknout. Jednalo se především o možnost změny jednotek a zamknutí os při pohybu. Problém byl vyřešen přidáním tooltipů[2] ke všem tlačítkům, kterých se toto týkalo.

Někteří testeři měli problém s orientací ve scéně; osám totiž chyběly popisky a proto nebylo jasné kterou klávesovou zkratku použít, když uživatel chtěl rotovat okolo konkrétní osy nebo některou osu uzamknout. Jednou mož-

[1] Mám tím na mysli chyby v GUI. Některé větší chyby se vyskytují v jádře (viz kapitola Nedostatky v jádře).

[2] Krátký popisek, který se objeví, pokud na tlačítko najedeme myší.

ností bylo přidat každé ose popisek ke kraji pohledu. Toto řešení není ovšem uživatelsky přívětivé, protože při rotacích s osami by popisky moc měnily svoje pozice a uživatel by je musel pokaždé na scéně hledat. Zvolila jsem proto nakonec jiné řešení: do pravého horního rohu scény byla přidána popsaná miniatura os kartézského souřadného systému, která odpovídá aktuálnímu natočení os ve scéně.

Poslední obtíže způsobovalo vytvoření záznamu na holografické desce. Po kliknutí na tlačítko *Vytvořit záznam* byl uživatel vyzván dialogovým oknem k výběru referenčního objektu podle jeho jména. Jméno objektu si ovšem uživatel může zjistit pouze v postranním panelu po kliknutí na daný objekt (pokud si jej nezapamatuje předem). Kvůli chybě v jádře (viz kapitola Nedostatky v jádře) není možné toto okno zavřít a uživatel byl tak nucen záznam vytvořit, zjistit si jméno požadovaného objektu a vytvořit záznam znovu. Implementovala jsem tedy možnost si v globálním nastavení zapnout vykreslení jmen objektů na pracovní plochu.

7 Nedostatky v jádře

Při implementaci GUI jsem narazila na několik chyb, které vznikají v jádře aplikace. Přestože předmětem této bakalářské práce je pouze GUI, povedlo se mi některé z těchto chyb opravit. Nejprve popíši ty opravené (co chybu způsobovalo a jak jsem provedla nápravu) a poté ostatní.

Opravené chyby

- Při kopírování objektů mezi více stoly nebyl právě vytvořené kopii nastaven nový stůl. To způsobovalo u zkopírovaných objektů nemožnost je mazat a při pohybu objektem se neaktualizovaly správně paprsky.
- Ke třídě Shape[1] úplně chybělo rozhraní pro komunikaci GUI s jádrem. Rozhraní jsem doplnila.
- Při některých změnách vlastností objektů se neaktualizovaly příslušné paprsky kvůli několika chybějícím volání metody, která se o tuto aktualizaci stará.
- V celém jádře zcela chyběly jednotky, ve kterých se mají zobrazovat objekty. Doplnila jsem tedy třídu `Units`, která obsahuje jednotky pro pozici, velikost a úhly natočení; pro plošné světlo navíc obsahuje jednotky pro šířku svazku. Úhly je možné přepínat mezi stupni a radiány, ostatní lze přepínat mezi milimetry, centimetry, decimetry a metry.

Neopravené chyby

- U bodového světla nelze korektně nastavit vrcholový úhel větší než 180°.
- Při rekonstrukci hologramu se občas objeví paprsek vedoucí do bodu [0,0] (levý horní roh pohledu).
- Při vytváření záznamu nelze zavřít okno, ve kterém se vybírá referenční objekt. Toto by obvykle byl problém GUI, nicméně autor jádra si vytváření tohoto okénka přesunul do jádra aplikace. Na návratových hodnotách tohoto dialogu je přímo závislá část jádra, a proto není bez větších komplikací možné vrátit dialogové okénko do GUI.

[1] Třída obsahuje seznamy stěn a vrcholů objektu nezbytné pro vykreslení.

8 Závěr

Hlavním cílem této bakalářské práce bylo navrhnout a implementovat nové grafické uživatelské rozhraní s podporou 3D zobrazení k aplikaci pro simulaci holografie, která byla v minulých letech vyvíjena v rámci jiných bakalářských prací a jejíž výpočetní jádro nyní vytvářel Kamil Rendl v rámci své diplomové práce. Dále bylo nutné grafické rozhraní řádně otestovat (především uživatelsky) a případné chyby opravit.

Cíle se podařilo splnit – bylo navrženo a implementováno zcela nové rozhraní, které v mnoha ohledech poskytuje podobnou funkčnost jako GUI původní aplikace, ovšem nově pracuje ve 3D prostoru. Důraz byl kladen především na správné zobrazení v tomto prostoru – posuny a rotace objektů, jejich viditelnost nebo projekce do 2D. Úspěšná byla také snaha implementovat GUI tak, aby bylo snadno a rychle rozšiřitelné o další optické členy s co nejmenšími úpravami zdrojového kódu.

Kromě vlastního testování správného 3D zobrazení bylo GUI testováno také uživatelsky, především co se týče ovládání, přístupnosti a přehlednosti. V těchto oblastech testeři odhalili několik chyb či nedostatků, ovšem všechny byly záhy odstraněny. Několik chyb, týkajících se funkčnosti aplikace, bylo nalezeno taktéž v jádře aplikace, z nichž některé byly nad rámec této bakalářské práce opraveny.

Literatura

[1] RENDL, Kamil. *Vizualizace principu hologramu*. Plzeň, 2012. Bakalářská práce. Západočeská univerzita v Plzni.

[2] HADÁČEK, Michael. *Vizualizace principu hologramu*. Plzeň, 2014. Bakalářská práce. Západočeská univerzita v Plzni.

[3] RENDL, Kamil. *Simulátor optických jevů*. Plzeň, 2015. Diplomová práce. Západočeská univerzita v Plzni.

[4] ŽÁRA, Jiří et al. *Moderní počítačová grafika*. 2., přeprac. a rozš. vyd. Brno: Computer Press, 2004. 609s. ISBN 80-251-0454-0.

[5] SLABAUGH, Gregory G. *Computing Euler angles from a rotation matrix* [online], 1999 [cit. 2016-04-12]. Dostupné z: `http://staff.city.ac.uk/~sbbh653/publications/euler.pdf`

[6] How to Use CardLayout. In: *The Java Tutorials* [online]. [cit. 2016-04-12]. Dostupné z: `http://da2i.univ-lille1.fr/doc/tutorial-java/uiswing/layout/card.html`

A Uživatelská příručka

A.1 Překlad a spuštění

Aplikace je na přiloženém CD dodána jako spustitelný `JAR` soubor, přeložený na `JDK` verze 1.8. Pro jeho spuštění je tedy nutné mít `JRE` této verze nainstalované. V případě, že je na počítači Java pouze starší verze (nutně ovšem včetně `JDK`), obsahuje CD taktéž zdrojové soubory a `ANT` skript, kterým lze tyto soubory zkompilovat a zabalit do spustitelného `JAR` souboru.

Aplikaci lze spustit (ve složce, kde se nachází `JAR` soubor) příkazem

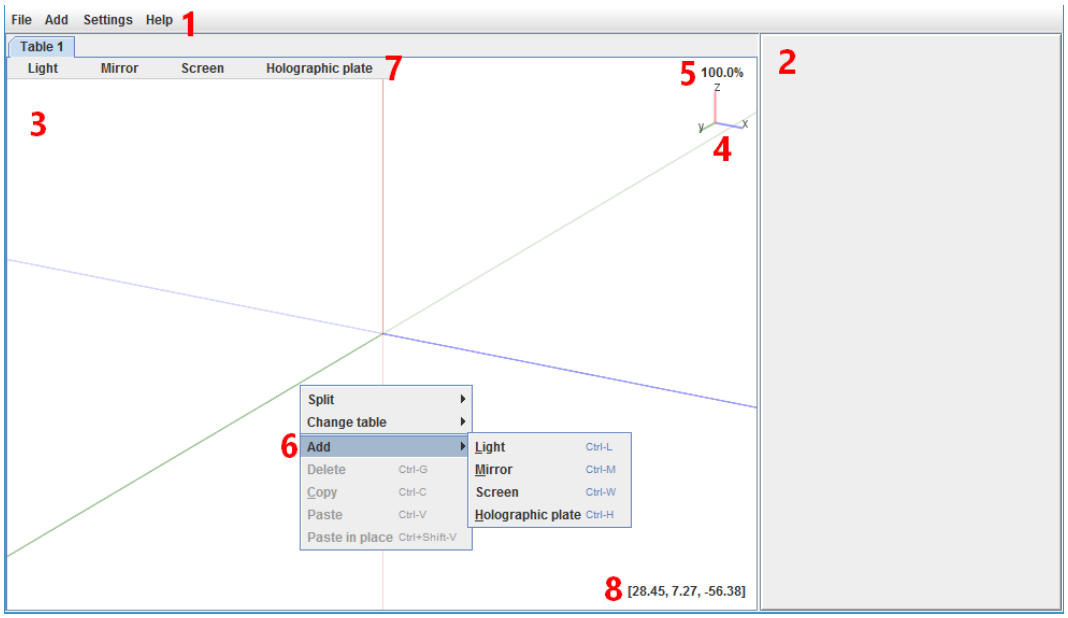
```
java -jar OpticalSimulator.jar
```

nebo dvojklikem na tento soubor. Případné přeložení `ANT` skriptem lze provést příkazem `ant` ve složce, kde se nachází soubor `build.xml`.

A.2 Seznámení s oknem

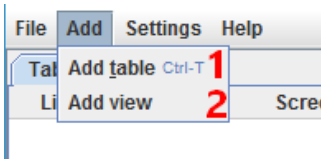
Po spuštění se uživateli zobrazí hlavní okno aplikace. To je rozděleno na tři části: v horní části se nachází menu **A.1 (1)**, po pravé straně je panel pro nastavení objektů **A.1 (2)** a v levé části, která zaujímá nejvíce prostoru, se nachází pracovní plocha **A.1 (3)**, na kterou se při simulaci umisťují optické objekty.

Stoly představují jednu simulaci, ty můžeme přidat pomocí menu pod položkou Add – Add Table **A.2a (1)**. Na každou simulaci se můžeme dívat z různých úhlů natočení – k tomu slouží tzv. *pohledy*. Pohledy můžeme vytvořit opět přes menu pod položkou Add – Add View **A.2a (2)**. Pracovní plochu je dále možné různě dělit na menší části a tím taktéž vytvářet nové pohledy. Toho docílíme kliknutím pravým tlačítkem na pracovní plochu (do pohledu, který chceme rozdělit) a vybráním položky Split – Split horizontally (resp. Split vertically) **A.2b (1)**; to vytvoří nový pohled na tentýž stůl (ve stejné položce menu je možné jej i smazat). Pokud bychom chtěli mít na jedné pracovní ploše více simulací, docílíme toho opět vyvoláním kontextového menu pravým

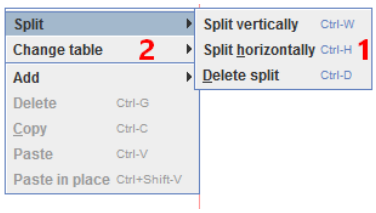


Obrázek A.1: Okno aplikace po spuštění.

kliknutím na požadovaný pohled a vybráním položky Change table **A.2b (2)**, kde následovně vybereme jméno stolu, který zde chceme zobrazit.



(a) Menu pro přidání stolu a pohledu.



(b) Menu pro rozdělování pohledů a změnu stolu.

Obrázek A.2: Menu aplikace

A.3 Možnosti pracovní plochy

Ovládání scény

Pracovní plocha po spuštění zobrazuje jeden stůl s vykreslenými osami KSS. Modrou barvou je znázorněna osa x, zelenou osa y, červenou osa z; barvy jsou vždy sytější na kladné poloose oproti poloose se zápornými hodnotami. Pro lepší orientaci aktuálního natočení os slouží miniatura umístěna v pravém horním rohu pracovní plochy **A.1 (4)**.

Výchozím zobrazením je pohled na rovinu určenou osami x a z, to lze ovšem rychle změnit klávesami Q, W a E, které přepnou pohled do rovin XY, XZ a YZ. Pohledy je možné otáčet vždy okolo jedné z os x, y a z pomocí klávesových zkratek `CTRL+A`, `CTRL+S` a `CTRL+Z`. Pro opačný směr rotace slouží zkratky `CTRL+ALT+A`, `CTRL+ALT+S` a `CTRL+ALT+Z`.

Simulátor dále nabízí možnost aktuální pohled přibližovat, resp. oddalovat. To provedeme kolečkem myši (taktéž možné uskutečnit klávesami `+` a `-`), přičemž střed KSS zůstane na místě. Pokud navíc držíme klávesu `SHIFT`, zůstane na místě bod, kde je umístěn kurzor myši. V pravém horním rohu **A.1 (5)** se zobrazují procenta aktuálního přiblížení; kliknutím na tyto procenta se scéna vrátí do výchozího měřítka `100%`.

Pohled lze taktéž posunout držením klávesy `SHIFT` a následným stisknutím a táhnutím myši (Drag and Drop).

Akce nad objekty

Objekty lze na scénu přidat kliknutím pravým tlačítkem myši na požadované místo a následným vybráním daného objektu pod položkou Add **A.1 (6)**. Druhou možností je menu rychlého přístupu v levém horním rohu každého pohledu **A.1 (7)**. Po vybrání požadovaného objektu pak uživatel vloží tento objekt na místo, kam klikne (souřadnice kurzoru jsou zobrazeny vždy v aktuálním pohledu v pravém dolním rohu **A.1 (8)**). Pokud ale tlačítko stiskneme za držení klávesy SHIFT, tlačítko se zamkne a vkládáme tento typ objektu tak dlouho, dokud tlačítko nedeaktivujeme opětovným kliknutím. Objekty je pak možné posouvat pomocí Drag and Drop. Rotace okolo os x, y a z provádíme taktéž pomocí Drag and Drop, ale za držení kláves `A`, `S`, a `D` (směr rotace určuje to, zda myší táhneme dolů nebo nahoru).

A.4 Panel nastavení

V panelu nastavení (pravá strana okna) je vždy zobrazen aktuálně vybraný prvek; výběr prvku probíhá tak, že na něj klikneme na pracovní ploše. Za aktuálně vybraný prvek se ovšem přednostně považuje ten, který byl právě přidán na plátno. Tento prvek zůstane vybrán, dokud není přidán na plátno nový prvek, nebo nějaký stávající vybrán kliknutím.

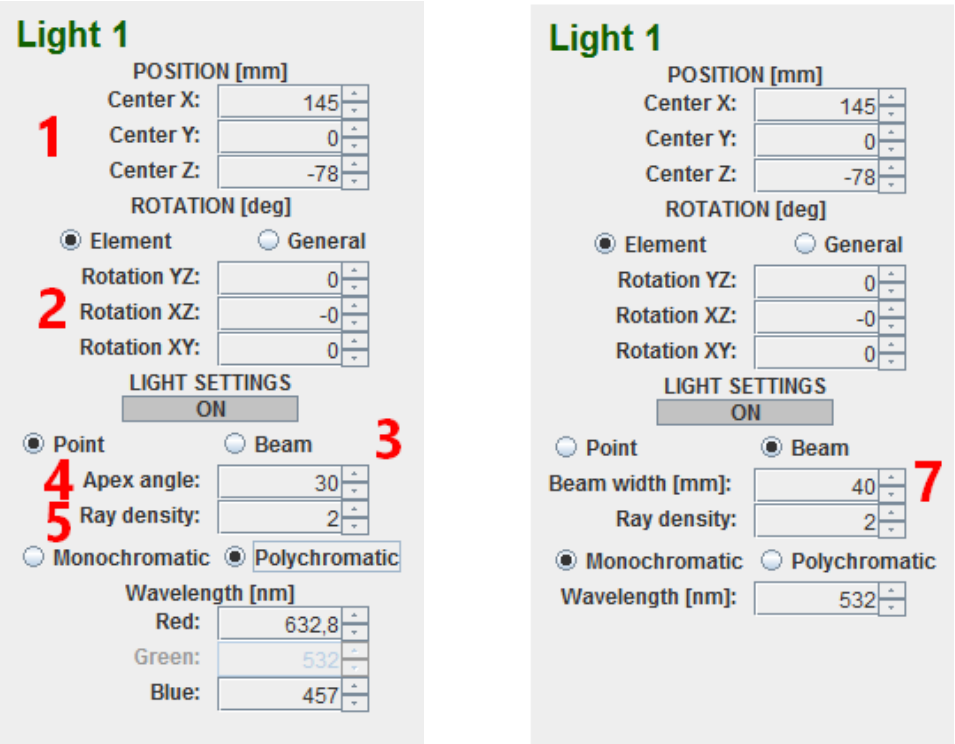
V tomto panelu nastavujeme objektům vlastnosti a parametry, přičemž některé z nich mají všechny typy optických objektů společné. Jsou to tyto:

- Pozice objektu na scéně, resp. souřadnice KSS **A.3a (1)**.
- Natočení objektu vzhledem k osám KSS **A.3a (2)**.
- Uzamknutí posunu objektu ve směru některé souřadnice (např. chcemeli optickou soustavu vytvářet pouze v jedné rovině) tak, že klikneme na popisek souřadnice.
- Změna jednotek, ve kterých se v panelu zobrazuje pozice objektu a úhel natočení. Provedeme kliknutím na text jednotek u pozice, rotace a velikosti.

Ostatní vlastnosti a nastavení jsou speciální vždy pro nějakou podmnožinu typů objektů.

Světlo

Světelný zdroj lze přepínat mezi bodovým a plošným typem **A.3a (3)**. Dále je světlu (oběma typům) možné nastavit, jestli má být monochromatické nebo polychromatické a příslušné vlnové délky **A.3a (6)** (Kliknutím na popisky vlnových délek polychromatického světla můžeme zapnout, resp. vypnout jednotlivé složky). Dalším nastavením je počet paprsků, které světlo vyzařuje **A.3a (5)**. Speciálně bodovému světlu pak navíc nastavujeme vrcholový úhel **A.3a (4)**, plošnému světlu šířku svazku **A.3b (7)**.



(a) Plošné světlo (b) Bodové světlo

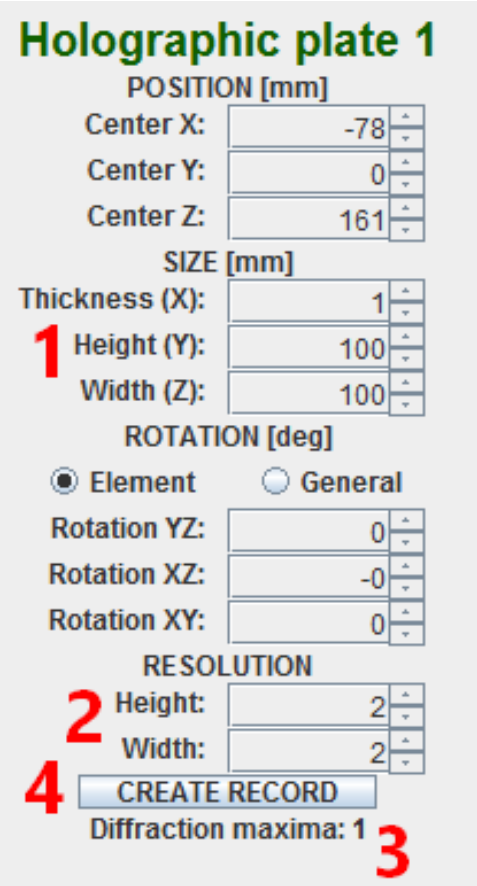
Obrázek A.3: Panel nastavení světla

Zrcátko, stínítko

Těmto objektům lze navíc nastavit pouze rozměry **A.4 (1)**.

Holografická deska

Holografické desce také můžeme měnit rozměry **A.4 (1)**, ale navíc ještě rozlišení **A.4 (2)** a difrakční maxima (dialogové okno se zobrazí po dvojkliku na popisek) **A.4 (3)**. Dále na ní můžeme vytvořit záznam **A.4 (4)**.

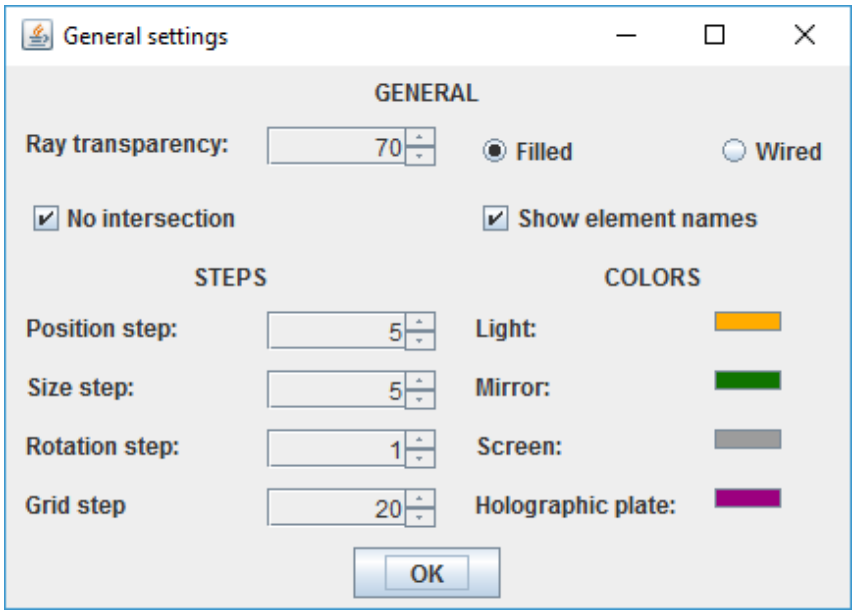


Obrázek A.4: Panel nastavení pro holografickou desku.

A.5 Globální nastavení

V hlavním menu je uživateli přístupné globální nastavení aplikace (viz obr. **A.5**).

- `Ray transparency` – nastavení průhlednosti paprsků v procentech
- `Filled / Wired` – možnost `Filled` zobrazuje objekty s normální viditelností; `Wired` vykresluje scénu v drátěném módu
- `No intersection` – zaškrnuté vykresluje všechny paprsky; nezaškrtnuté nevykresluje paprsky, které nemají průnik s žádným objektem



Obrázek A.5: Okno aplikace po spuštění.

- `Show element names` – po zaškrtnutí se vykreslují do scény i jména objektů
- `STEPS:`
  - `Position step` – nastaví krok poli pro změnu pozice objektu v panelu nastavení
  - `Size step` – nastaví krok poli pro změnu velikosti objektu v panelu nastavení
  - `Rotation step` – nastaví krok poli pro změnu natočení objektu v panelu nastavení; současně se jedná o krok, o který se rotuje pomocí klávesových zkratek
  - `Grid step` – kliknutím na popisek zapneme pohyb objektů po mřížce; nastavená hodnota pak odpovídá hustotě mřížky (v milimetrech)
- `COLORS` – kliknutím na konkrétní barevnou kolonku se zobrazí dialogové okno pro výběr barvy daného objektu

A.6 Shrnutí ovládání a klávesových zkratek

Nyní shrnu všechny klávesové zkratky, kterých lze využívat v aplikaci. Budu používat zkratku DND, kterou myslím pohyb Drag and drop, tedy kliknutí a táhnutí myší.

- `CTRL + T` – přidání nového stolu
- `SHIFT + DEL` – smazání aktuální záložky
- `DEL` – smazání aktuálně vybraného objektu
- *Posun, rotace a přibližování scény*
  - `SHIFT + DND` – posun celé scény
  - `CTRL + A` – rotace okolo osy X
  - `CTRL + ALT + A` – rotace okolo osy X v opačném směru
  - `CTRL + S` – rotace okolo osy Y
  - `CTRL + ALT + S` – rotace okolo osy Y v opačném směru
  - `CTRL + D` – rotace okolo osy Z
  - `CTRL + ALT + D` – rotace okolo osy Z v opačném směru
  - `kolečko myši`, nebo klávesy `+ / -` – přibližují / oddalují pohled za středem KSS
  - `SHIFT + kolečko myši` – přibližuje / oddaluje pohled za aktuálně umístěným kurzorem myši
- *Přepínání rovin scény*
  - `Q` – přepne do roviny XY
  - `CTRL + Q` – přepne do roviny XY z opačné strany
  - `W` – přepne do roviny YZ
  - `CTRL + W` – přepne do roviny YZ z opačné strany
  - `E` – přepne do roviny XZ
  - `CTRL + E` – přepne do roviny XZ z opačné strany
- *Posun objektů* – všechny následující funkce lze využít, když je myš nad požadovaným objektem (objekt se modře zbarví).

  - `DND` – posun objektu v aktuální rovině natočení
  - `X + DND` – posun objektu se zamknutou souřadnicí X
  - `Y + DND` – posun objektu se zamknutou souřadnicí Y
  - `Z + DND` – posun objektu se zamknutou souřadnicí Z

- *Rotace objektů* – všechny následující funkce lze využít, když je myš nad požadovaným objektem (objekt se modře zbarví). Směr rotace je určen změnou souřadnice kurzoru na ose y (svislá souřadnice obrazovky).
  - `A + DND` – rotace objektu okolo jeho osy X
  - `S + DND` – rotace objektu okolo jeho osy Y
  - `D + DND` – rotace objektu okolo jeho osy Z

- *Přidávání objektů*
  - `CTRL + L` – přidá světlo tam, kde je aktuálně umístěn kurzor
  - `CTRL + M` – přidá zrcátko tam, kde je aktuálně umístěn kurzor
  - `CTRL + R` – přidá stínítko tam, kde je aktuálně umístěn kurzor
  - `CTRL + H` – přidá holografickou desku tam, kde je aktuálně umístěn kurzor

- *Přidávání objektů* – menu rychlého přístupu (levý horní roh každého pohledu)
  - kliknutí na požadované tlačítko – kliknutí na daný pohled přidá objekt vybraného typu (právě 1×)
  - `SHIFT` + kliknutí na požadované tlačítko – zamknutí tlačítka, klikáním na daný pohled přidáváme objekt vybraného typu tak dlouho, dokud tlačítko opětovným kliknutím nedeaktivujeme

- *Kopírování a vkládání objektů*
  - `CTRL + C` – zkopírování aktuálně vybraného objektu do schránky
  - `CTRL + V` – vložení objektu ze schránky na místo, kde je aktuálně umístěn kurzor
  - `CTRL + SHIFT + V` – vložení objektu ze schránky na místo, kde byl původně objekt umístěn

B Testovací scénář

- Záznam hologramu:
  - Na pracovní plochu přidejte 2 světla, jedno bodové, jedno plošné. Přidejte ještě záznamový materiál (holografickou desku).
  - Objektům nastavte pozici a natočení tak, aby obě světla osvětlovala holografickou desku.
  - Pracovní plochu libovolně rozdělte, v alespoň jednom pohledu změňte natočení scény (klávesovou zkratkou `CTRL+A` (resp. `CTRL+S`, `CTRL+D`) pro otočení okolo os x, y a z). Scénu lze i přibližovat/oddalovat kolečkem myši, nebo celou posunovat táhnutím myši za stisknutí klávesy `SHIFT`. Pro přepnutí do základních natočení (roviny `XY`, `XZ` a `XZ`) lze využít kláves `Q`, `W` a `E`.
  - Světlům zvyšte hustotu paprsků na alespoň 4, holografické desce zvyšte rozlišení taktéž na alespoň 4 (výšku i šířku).
  - Zjistěte si jméno plošného světla (po kliknutí na toto světlo uvidíte jméno světla v postranním panelu; dvojklikem na toto jméno je možné jej přejmenovat). Na holografické desce vytvořte záznam a vyberte podle jména plošné světlo.
- Rekonstrukce hologramu:
  - Vytvořte nový stůl.
  - Na tento stůl zkopírujte holografickou desku, na které jsme právě vytvořili záznam.
  - Přidejte na stůl světlo tak, aby osvětlovalo holografickou desku.
  - Pracovní plochu rozdělte vodorovně a na jednom z pohledů přepněte na původní stůl (kliknutím pravým tlačítkem myši do požadovaného pohledu), na kterém jsme hologram zaznamenávali.
  - Nyní můžeme pozorovat zrekonstruovaný hologram. Vyzkoušejte si, co se děje, když například světlu měníme různé vlastnosti (pozici, natočení, vlnovou délku, atd.).

C Obsah CD

- `aplikace`
  - `doc` – vygenerovaný Javadoc
  - `src` – zdrojové soubory aplikace
  - `OpticalSimulator.jar` – spustitelná verze aplikace
- `text`
  - `BP.pdf` – text ve formátu pdf
  - `src` – zdrojové soubory textu
- `README.txt` – obsah CD