FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Název:	Testování užitečnosti nástroje pro výuku transformací
Student:	Vít Zadina
Vedoucí:	Ing. Petr Felkel, Ph.D.
Studijní program:	Informatika
Studijní obor:	Webové a softwarové inženýrství
Katedra:	Katedra softwarového inženýrství
Platnost zadání:	Do konce letního semestru 2019/20

Pokyny pro vypracování

Seznamte se s grafickým interaktivním nástrojem na výuku transformací (i3t), který je vyvíjen na katedře počítačové grafiky a interakce, a s výsledky testování jeho uživatelského rozhraní.
Vyjděte z existujícího návrhu nového rozhraní, který metodou user centered design postupně dotvořte do nového návrhu uživatelského rozhraní. Svůj postup patřičně dokumentujte.
Spolupracujte na návrhu postupů pedagogického testování získaných vědomostí za pomoci i3t.

Seznam odborné literatury

Petr Felkel, Alejandra Magana, Michal Folta, Alexa Gabrielle Sears, Bedrich Benes: 3T: Using Interactive Computer Graphics to Teach Geometric Transformations.*Eurographics Education Papers 2018. https://www.eurographics2018.nl/program/education-papers/*

Michal Folta: Teaching of Transformations, Masters theses. *Department of Computer Graphics and Interaction, CTU Prague. 2018 https://dcgi.fel.cvut.cz/theses/2016/foltamic*

Lukáš Pilka: *Grafické návrhy rozhraní pro nástroj I3T*, interní komunikace, 2018

Ing. Michal Valenta, Ph.D.
vedoucí katedry

doc. RNDr. Ing. Marcel Jiřina, Ph.D.
děkan

V Praze dne 4. února 2019

FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

Bakalářská práce

Testování užitečnosti nástroje pro výuku transformací

Vít Zadina

Katedra softwarového inženýrství - zaměření počítačová grafika
Vedoucí práce: Ing. Petr Felkel. Ph.D.

12. května 2019

Poděkování

Rád bych poděkoval svému vedoucímu Ing. Petru Felkelovi, Ph.D. za jeho cenné rady, trpělivé a odborné vedení a gramatické i formální úpravy. Dále bych rád poděkoval panu doc. Ing. Janu Schmidtovi, Ph.D. za rady a konzultace ohledně možností testování mého návrhu. Dále bych rád poděkoval Bc. Ondřeji Brémovi za pomoc při uživatelském testování v laboratoři. Děkuji panu prof. Bedřichu Benešovi za rozeslání dotazníku na universitě v Purdue. Nakonec bych rád poděkoval všem účastníkům jednotlivých testů za pomoc při vývoji nového uživatelského prostředí.

V neposlední řadě bych rád poděkoval své rodině, všem svým blízkým a přátelům, za jejich podporu a přízeň.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval(a) samostatně a že jsem uvedl(a) veškeré použité informační zdroje v souladu s Metodickým pokynem o etické přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. V souladu s ust. § 46 odst. 6 tohoto zákona tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou, a veškeré jejich dokumentace (dále souhrnně jen „Dílo"), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla, a za jakýmkoli účelem (včetně užití k výdělečným účelům). Toto oprávnění je časově, teritoriálně i množstevně neomezené. Každá osoba, která využije výše uvedenou licenci, se však zavazuje udělit ke každému dílu, které vznikne (byť jen zčásti) na základě Díla, úpravou Díla, spojením Díla s jiným dílem, zařazením Díla do díla souborného či zpracováním Díla (včetně překladu), licenci alespoň ve výše uvedeném rozsahu a zároveň zpřístupnit zdrojový kód takového díla alespoň srovnatelným způsobem a ve srovnatelném rozsahu, jako je zpřístupněn zdrojový kód Díla.

V Praze dne 12. května 2019 ……

České vysoké učení technické v Praze
Fakulta informačních technologií

Odkaz na tuto práci

Zadina, Vít. *Testování užitečnosti nástroje pro výuku transformací.* Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2019.

Abstrakt

Transformace jsou v počítačové grafice velmi důležité, proto vznikl program *I3T* pro jejich výuku. Bohužel program má zastaralou grafiku a špatné ovládání – uživatel se v něm nedokáže bez návodu orientovat. Proto jsem analyzoval uživatelské rozhraní a funkce nástroje *I3T*. Na základě analýzy jsem postupně vytvořil nové návrhy uživatelského rozhraní a nových funkcí programu (tutoriál, rozložení programu na více oken, okno pro přiblížení matice, ...). V programu *Axure* vznikl prototyp, který tyto návrhy spojil. Prototyp byl postupně testován pomocí pluralistického testování, kognitivního průchodu, heuristické analýzy a uživatelského testování. Na základě výsledků testů byly provedeny úpravy prototypu a vylepšeny navrhované funkce. Součástí práce je dotazník zabývající se nejčastěji používanými klávesovými zkratkami v 3D programech. Na základě analýzy dotazníku byly navrženy, i pro program *I3T*, nové klávesové zkratky. Výsledkem této práce je otestovaný prototyp (webová stránka) uživatelského prostředí pro program na výuku transformací *I3T*, který odstranil 90 % nedostatků kritizovaných v současné verzi programu *I3T*, ale i v podobných aplikacích. Popis nových nebo změněných funkcí a hlavně prototyp je podkladem pro reimplementaci programu *I3T*, která na jeho základě probíhá. Pro většinu dotazovaných lidí je nové uživatelské prostředí na první pohled intuitivnější a graficky lepší.

Klíčová slova program pro výuku 3D transformací, I3T, návrh uživatelského rozhraní, testování uživatelského rozhraní, 3D transformace, výuka transformací, počítačová grafika, klávesové zkratky

Abstract

Transformations in computer graphics are very important, therefore was created a program *I3T* for their teaching. Unfortunately, the program had an outdated graphics and a poor control – user is not able to work in the program without instructions. That's the reason why I analyzed a user interface and features of the program *I3T*. Based on an analysis I created a new user interface designs and new features of the program (tutorial, the layout of the program to multiple windows, zoom window, ...). In the *Axure* was created a prototype, which merged the designs. The prototype was tested by Pluralistic Walkthrough, Cognitive Walkthrough, Heuristic analysis and user testing. Based on results of tests the prototype was redesigned and the features were improved. A part of the thesis is a questionnaire with the most used keyboard shortcuts in 3D programs. Based on the analysis of the questionnaire were designed new shortcuts also for the *I3T* program. The result of this thesis is a tested prototype (website) of the user interface of the program for teaching transformations *I3T*. The program removed 90 % errors criticized in the current version of the *I3T* program, but also in similar applications. A description of the new or changed features and mainly the prototype is a basis for a reimplementation *I3T*, which works on a basis of it. For the major part of people, which attended the testing, the interface is graphically better and more intuitive.

Keywords tool for teaching transformations, I3T, user interface design, testing user interface, 3D transformations, teaching transformations, computer graphics, keyboard shortcuts

Obsah

Seznam obrázků

Seznam tabulek

Úvod

Lineární transformace jsou v počítačové grafice velmi důležité. Proto se učí ve všech předmětech počítačové grafiky po celém světe, ale studenti s nimi mají velmi často velké potíže.

V roce 2016 byl na fakultě elektrotechnické na ČVUT jako magisterská práce Michala Folty vytvořen program *I3T* [1] pro výuku 3D transformací, který má pomoc při studiu počítačové grafiky po celém světě. Bohužel aktuální uživatelské rozhraní včetně klávesových zkratek je velmi špatně navrženo a studenti program téměř nevyužívají.

Cílem práce je analyzovat s čím mají studenti v programu *I3T* problémy a jaké nové funkce by v programu uvítali. Prozkoumat aktuální možnosti výuky transformací. Provést analýzu uživatelského prostředí grafických aplikací a na jejím základě navrhnout nové uživatelské rozhraní programu *I3T*.

Doufám, že novým uživatelským rozhraním program *I3T* studenty zaujme a díky nové funkci tutoriálu se jej velmi rychle naučí používat a získají nový způsob učení. Změna klávesových zkratek uživatelům ulehčí ovládání programu a budou se tak moci věnovat problémům s transformacemi místo zkoumání ovládání programu *I3T*.

Jednotlivé návrhy budou otestovány a implementovány do prototypu nového uživatelského prostředí programu *I3T*. Prototyp bude dále testován a vyvíjen, poté bude sloužit vývojovému týmu jako vzor při vývoji nové verze programu *I3T*.

Kapitola 1

Cíle práce

Cílem práce je analyzovat funkce programu pro výuku lineárních transformací *I3T* a na základě této analýzy navrhnout nové uživatelské rozhraní programu. Analyzovat požadavky studentů v rámci předmětu počítačová grafika (*BI-PGR*) a co studenti na probírané látce nechápou.

Dílčím cílem je prozkoumat, jestli studenti počítačové grafiky výše zmíněný nástroj používají a v jakém rozsahu. Prozkoumat další možnosti studia lineárních transformací, které studenti počítačové grafiky využívají. Vytvořit uživatelsky přívětivé rozhraní, aby se uživatel dokázal velmi snadno orientovat v programu. Nynější uživatelské prostředí je v takovém stavu, že většina uživatelů nenalezne ani předpřipravené tutoriály a celkové ovládání programu má dost nedostatků (nestandardní klávesové zkratky, problémové názvy a umístění vstupů a výstupů, nestandardně fungující výběr, ...).

Dílčím cílem je, na základě analýzy vytvořit návrh, který poté porovnám s návrhem z katedry počítačové grafiky a interakce FEL ČVUT v Praze. Spojením obou návrhů vytvořit základní vzhled a vědět, jaké funkce by měl nástroj mít. Aktuální předpřipravené funkce přetransformovat do jednotlivých tutoriálů, které se zobrazí při spuštění programu a vysvětlí probíranou látku a naučí uživatele ovládat program.

Dílčím cílem je vytvoření prototypu, na kterém bude možné udělat různé testy nových návrhů, na základě kterých poznám, co lze zlepšit, co uživatele mate, a zjistím jaké funkce jsou v programu pro uživatele nejdůležitější. Hlavní potíží aktuální verze nástroje *I3T* bylo jeho složité ovládání, špatné rozložení klávesových zkratek a špatně zvolené názvy jednotlivých objektů/funkcí. Uživatelé se při tvorbě složitější scény ztratili a výuku v programu vzdali. Dílčím cílem je na základě průzkumu vytvořit nové klávesové zkratky pro jednotlivé funkce programu. Návrh nových funkcí a prototyp uživatelského rozhraní bude sloužit vývojovému týmu k reimplementaci programu *I3T*.

Dílčím cílem je spolupráce na návrhu postupů pedagogického testování získaných vědomostí za pomoci nástroje pro výuku transformací *I3T*.

Kapitola 2

Analýza

I3T je program pro výuku transformací v 3D prostoru. Program v aktuální verzi (pod tímto názvem se myslí verze programu vydána 22. 3. 2018, která je přiložena na CD, dále jen aktuální verze programu) obsahuje předpřipravené scény na různá témata týkající se 3D transformací, například scénu na gimbal lock, na perspektivní kameru, jednoduché scény na pohyby s jednotlivými objekty. Program umožňuje sestavovat graf scény pomocí sekvencí a interaktivně sledovat posun objektů při změnách transformací. Ke každé sekvenci se dá připojit (*bind*) objekt, který ovlivňují matice v sekvenci a ve všech sekvencích směrem ke kořeni (sekvence zapojené napravo objekt neovlivní). U připravených scén je přidán text s popisem scény.

Každá sekvence má tři výstupy a to vstup pro propojování (násobení) jednotlivých sekvencí, dále vstup nazvaný sklad (*storage*), z kterého si může uživatel zobrazit obsah vybrané sekvence. Poté nepojmenovaný výstup, který předá matici, která představuje vynásobení všech matic připojených směrem ke kořeni. Tyto výstupy slouží hlavně pro operátory, které vytvářejí matice s jimiž můžete dále pracovat, případně se dají za pomoci nich upravovat jednotlivé položky a předávat dále.

V *I3T* lze i rozpohybovat/animovat objekty, toho lze dosáhnout za pomoci krabičky float cycle, která generuje hodnoty. Zapojením více krabiček *float cycle* do krabičky vektoru a při jejím spojení s dalšími operátory, tak lze vytvořit matici jejíž hodnoty se mění.

Program slouží pro výuku, ale v aktuální verzi se student moc informací ohledně teorie lineárních transformací nedočte a většinu věcí si musí studenti zkusit metodou pokus omyl nebo dohledat informace v jiných zdrojích.

2.1 S čím mají studenti předmětu počítačové grafiky největší problémy

Diskutoval jsem s absolventy předmětu PGR[1], abych zjistil, jak důležité je učivo transformací pro studenty v tomto předmětu. Dále jsem zjišťoval, s jakým učivem měli studenti problémy. Tyto informace budou využity při návrhu programu a pořadí implementace funkcí programu podle důležitosti.

Studenti vyjmenovali několik problémů a někteří i navrhli, jak by si představovali zobrazení tohoto problému v programu *I3T*.

- Umožnit zobrazení tělesa, aby byly vidět trojúhelníky, případně přehledný výpis bodů z tělesa (člověk by mohl posunout jeden bod a viděl by, jak se těleso změnilo), mohly by být vidět i koordináty textury, aby člověk viděl, jak se změní textura po změně parametru *u,v*.
- Zobrazení normál a možnost jejich editace.
- Co která transformace dělá a jak mají správně vypadat jednotlivé matice. (obsah jednotlivých matic, například, když chcete posunout objektem)
- Skládání transformací je sice pěkně popsané v přednášce, ale kdyby ho program nějak lépe vysvětlil, bylo by to lepší.
- Nastavování kamery (lookAt vektor, pohyb (*move*), přiblížení (*zoom*), rotaci (*rotation*), případně nějaký způsob jak kameru animovat v *I3T*, který by vygeneroval parametry do našeho kódu, např. zapnete animaci, přesunete objekt o pár polí dopředu –> vygeneruje se uživateli křivka pro pohyb objektu v závislosti na čase.
- Možnost rychlého vybrání či vytvoření gimbal locku s připojením teoretického základu, aby student tento problém z této ukázky jednoduše pochopil.
- Typy osvětlení a jejich nastavení, aby bylo vidět, co se změní přenastavením určitého parametru. Přidání různých světel do programu, změna jejich parametrů a pozorování jejich vlivu na vzhled scény. Světla si můžou studenti vyzkoušet i v různých 3D programech, ale zde nejsou k dispozici všechny parametry světel, které jsou potřeba pro jejich naprogramování, ani vysvětlení, jak jednotlivá světla fungují.
- Jak se mění mlha v závislosti na jednotlivých parametrech.

Program *I3T* již uživatelům umožňuje zobrazit obsah matic a jejich transformace, obsahuje i scénu zaměřenou na gimbal lock, bohužel u této scény není

[1] programování počítačové grafiky

dostačující popis tohoto problému, program zobrazuje normály objektů, bohužel není možná jejich editace a v programu lze přidat kameru i s maticemi pro její transformace, které uživatel bohužel musí postupně přidat sám (matice by se mohly vložit už s kamerou). Zobrazování mlhy a jejích parametrů by dle mého názoru mohlo být do programu přidáno, protože to není implementačně náročné. Automatická animace objektů a přidání světel do programu *I3T* je dle mého názoru nad rámec tohoto programu, studenti si tyto funkce mohou vyzkoušet v jiných 3D aplikacích.

2.2 Rešerše existujících nástrojů pro výuku transformací v 3D grafice

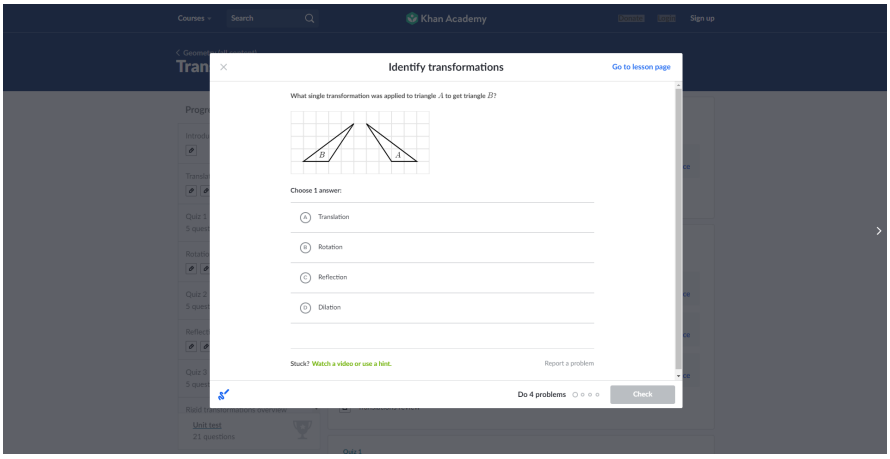
Způsobů pro výuku transformací existuje mnoho. Od knížek a webových stránek, které řeší čistě matematickou stránku, po interaktivní kurzy s videi nebo webovými stránkami s tutoriály, které vysvětlují teorii ohledně lineárních transformací interaktivně.

Programů věnující se této tématice je na světě mnoho. Většina těchto programů se zaměřuje na vysvětlení transformací v OpenGL nebo vysvětlují jen 2D lineární transformace. Nenašel jsem žádný program, který pracuje s 3D prostředím a umožňuje zobrazení jednotlivých transformačních matic. V následujících podkapitolách popíši zástupce jednotlivých způsobů výuky a jejich výhody a nevýhody.

2.2.1 Klasické teoretické základy

Vedle klasických zdrojů, kterými jsou různé učebnice lineární algebry, lze nalézt i knihy, které se věnují lineárním transformacím v 3D z pohledu herního vývojáře. Tyto knihy jsou vhodné pro programátory, protože zde je vysvětlena jak teoretická stránka, tak zde mohou nalézt i praktické příklady implementace. Tyto knihy neobsahují jen lineární transformace, ale všechny matematické základy potřebné pro tvorbu počítačové grafiky. Kniha, která je v internetových diskuzích nejčastěji doporučována pro začátečníky v počítačové grafice je 3D Math Primer For Graphics And Game Development [2].

Obsahují i knihy o programovacích jazycích, které obsahují kapitoly o lineárních transformacích a vysvětlují matematické základy společně s ukázkou kódu, který dané matematické problémy implementuje. Nejvíce programátorských knih se zmínkou o lineárních transformaci se zabývá knihovnou OpenGL. Pro příklad mohu uvést knihu 3D Computer Graphics: A Mathematical Introduction with OpenGL [3] a trochu novější knihu s ukázkami kódu ve WebGL Interactive Computer Graphics: A Top-Down Approach with WebGL (7th Edition) [4].



Obrázek 2.1: Kurz transformací od Khan Academy, je tvořen videi doplněnými texty s teorií a znalostními testy

2.2.2 Interaktivní výuka - videa, online kurzy

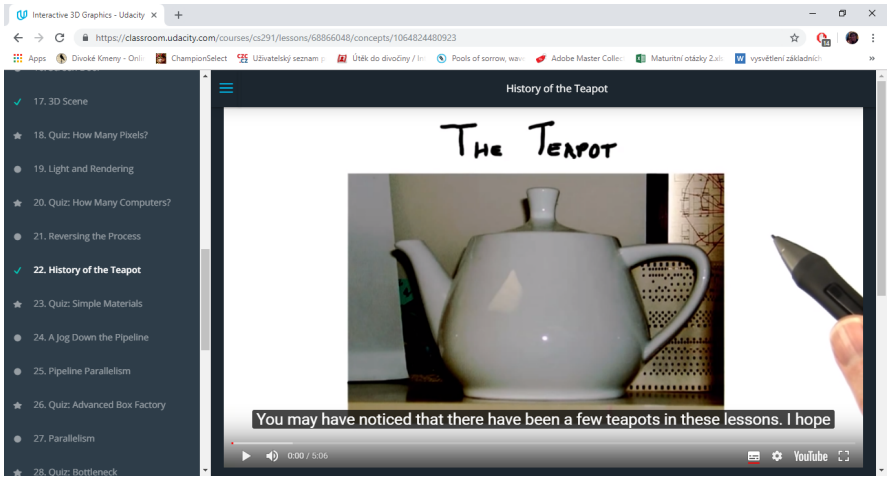
Při vyhledávání nástrojů na výuku transformací jsem nalezl několik youtube videí, kde lidé popisují 3D transformace, například [5]. Tato videa, ale neposkytnou více informací, než student získá na přednášce. Takže bych je nazval jako alternativní studijní materiál. Pro zopakování 3D transformací je skvělé video [6] o 3D transformacích.

Velmi zajímavý je kurz geometrických transformací Khan Academy [7]. Součástí kurzu jsou videa, která vám vysvětlí základy, doplněná o teoretické poznámky. U každého tématu si můžete právě naučenou látku vyzkoušet na příkladu, každá lekce je zakončena kvízem shrnující hlavní látku daných témat viz Obr. 2.1.

Tento kurz je vhodný pro začátečníky. Troufám si říci, že je se z něj každý schopen naučit základy geometrických transformací. Forma kurzu velmi motivuje žáka k učení. Testy nejsou těžké, jsou hodnocené hvězdičkami a student získává energii (skóre), která tento systém učení tvoří zábavnější. Dále jsem našel bezplatný kurz Interactive 3D Graphics viz Obr. 2.2, který má žáka naučit základy 3D grafiky a vývoje her. Kurz je v angličtině. Platforma, na které se kurz nachází, se jmenuje Udacity. Je zde hodně kurzů na mnoho témat. Na bezplatný kurz [8] navazuje další zpoplatněný kurz.

Kurz je tvořen sérií youtube videí, po každé kapitole následuje test. Po absolvování kurzu by student měl mít velmi dobré základy pro tvorbu 3D aplikací.

Velmi dobrým zdrojem informací, pokud chcete znalosti používat v programování, je webová stránka Tobyho Rufinuse OpenGL [9]. Jednotlivé operace s maticemi jsou doplněné obrázky. Druhou polovinu kurzu tvoří ukázky kódu. Pokud začínáte s *OpenGL*, tento web mohu velmi doporučit.



Obrázek 2.2: Grafické prostředí kurzu Udacity - Interactive 3D Graphics – video o vzniku konvičky (*teapot*) známé z mnoho 3D programů.

2.2.3 Programy pro výuku transformací

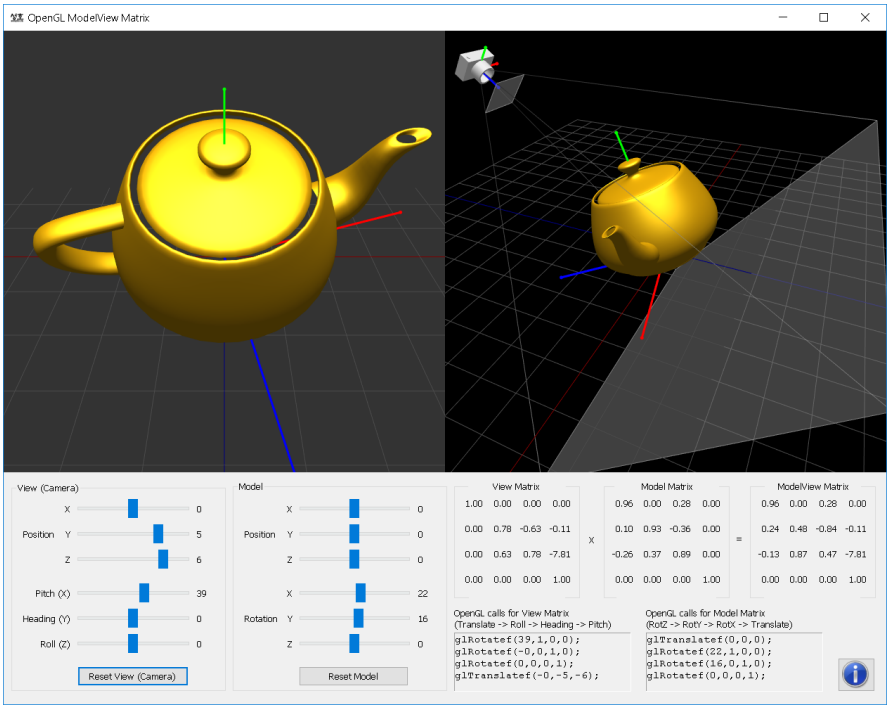
3D aplikaci určenou přímo pro výuku 3D transformací se mi bohužel nepovedlo najít. Pan Folta ve své magisterské práci zmiňuje a popisuje několik programů, z nichž jen dva pracují s 3D transformacemi a umí i zobrazovat matice. Bohužel ani jeden z nich neumí změnit pořadí matic. [10]

Prvním programem je *OpenGL ModelView Matrix* [11], který se nachází na osobních stránkách pana Song Ho Ahn a je součástí tutoriálu na 3D tranformace v *OpenGL*. Demo program viz Obr. 2.3 umožňuje nastavit pozici a orientaci kamery i modelu. Matice jsou zobrazovány v dolní části programu, bohužel nikde není objasněno jejich pořadí, ani není zmíněno, že pohledová matice je vlastně inverzí pohledové matice. Program ukazuje funkce OpenGL měnící rotaci a umístění kamery a objektu. Bohužel program používá starou verzi *OpenGL*, takže uvedené ukázky použitých funkcí nemají příliš vysokou pedagogickou hodnotu. [10]

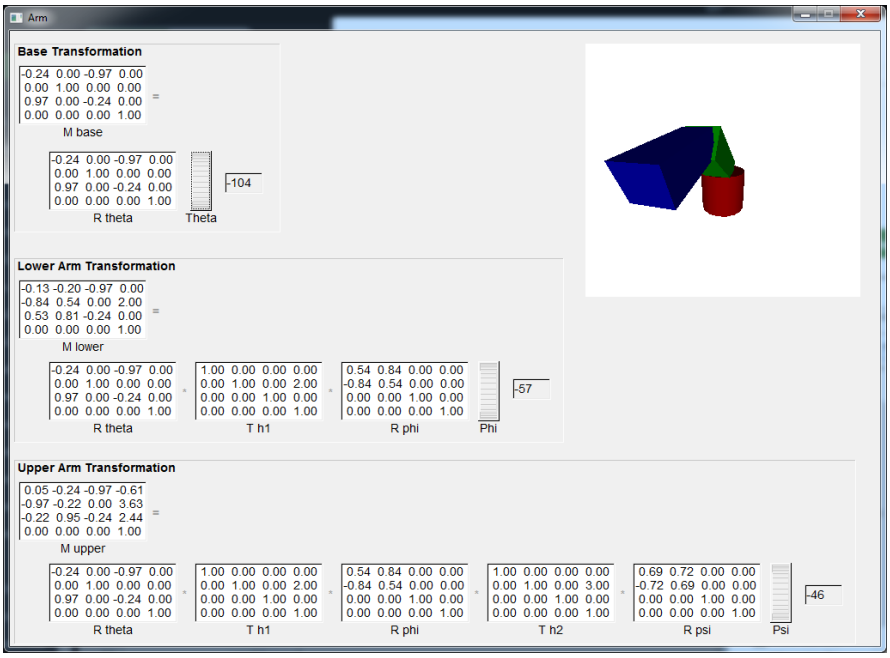
V roce 2018 byla vydána nová verze programu *OpenGL ModelView Matrix*. Změnilo se uživatelské prostředí a rozmístění matic. Logika programu nejspíše nebyla změněna. Program je dostupný na *Windows* a *macOS*.

Druhý program zmíněný v práci pana Folty se jmenuje *Arm* [13] a byl vytvořen na Waschingtonské univerzitě. Scéna se skládá z válce a dvou kvádrů, kterými může žák pomocí posuvníku rotovat. Uživatel sice vidí všechny matice, ale může měnit hodnotu jen tří matic, takže tento program slouží jako velmi jednoduchá ukázka 3D transformací viz Obr. 2.4.

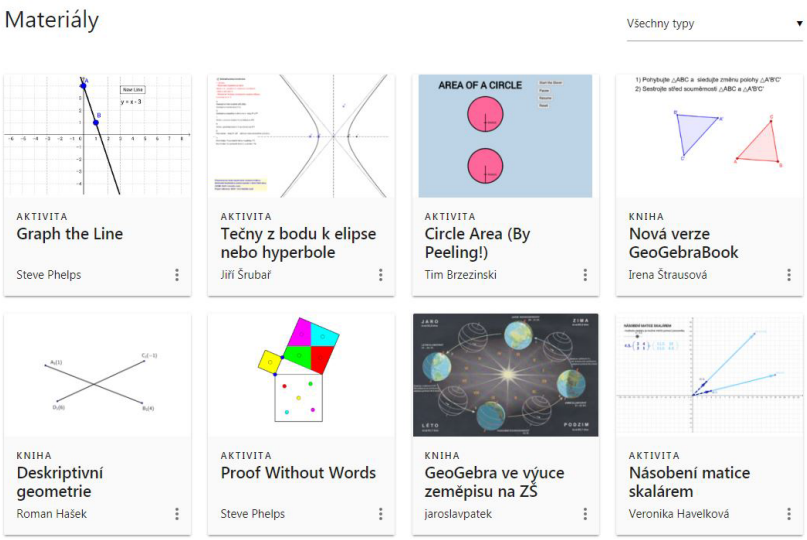
Geogebra je velmi rozsáhlý freeware program pro výuku matematiky [14]. Je dostupný ve webové verzi i jako desktopová aplikace, případně mobilní aplikace. Uživatelé sem přidávají materiály na různá matematická témata.



Obrázek 2.3: Demo program OpenGL ModelView Matrix [12]



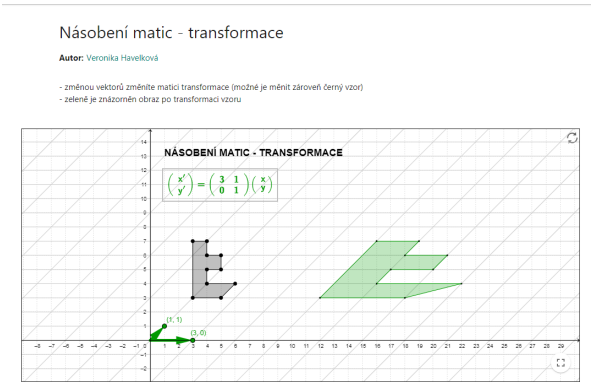
Obrázek 2.4: Arm - jednoduché demo sloužící k výuce na Washingontské univerzitě



Obrázek 2.5: Geogebra nabízí velké množství materiálů zabývajících se různými matematickými problémy

V jednotlivých tématech lze vyhledávat. Materiály tvoří sami uživatelé, je zde i několik materiálů v češtině viz Obr. 2.5. Pro ukázku funkcí Geogebry jsem si vybral téma Lineární algebra a geometrické transformace od Veroniky Havelkové [15].

V materiálu viz Obr. 2.6 jsou velmi dobře vysvětleny základní transformace. Vzhledem k tomu, že ukázky jsou interaktivní, může uživatel vzít bod, zatáhnout za něj a uvidí jak se obrazec změní. Pro zopakování lineární algebry, případně jako doplňkový učební materiál, je tato lekce velmi vhodná.



Obrázek 2.6: Geogebra – ukázka jedné části lekce, týkající se násobení matic. Je zde vidět, jak se změní obrazec, když změníte jednotlivé sloupce matice.

Celkově mohu portál Geogebra doporučit jako doplněk k učení matematiky. Jelikož je to webová aplikace, získáte odezvu okamžitě.

2.3 Co se nachází v I3T

Na největší nedostatek jsem narazil u bývalých studentů předmětu. Když jsem se jich ptal na jejich zkušenosti s *I3T*, někteří vůbec netušili, že program existuje. Ostatní o programu věděli, ale otevřeli ho jen několikrát. Z toho plyne, že program moc žáků reálně k učení nevyužívá.

Program by se mohl používat při vysvětlování teorie k danému cvičení. Velmi rychle studentovi ukáže, jak například funguje lookAt matice u kamery. V programu jsou připravená dema, která studentovi velmi rychle vysvětlí, jak který parametr změní scénu.

S programem se bez zhlédnutí manuálu/tutoriálu pracuje velmi obtížně. Do doby, než jsem se podíval na video popisující ovládání programu, bych do scény nepřidal ani kostku. Velmi snadno si uživatel může vyzkoušet práci s tělesem pomocí matic rotace (*rotation*), přesunu (*move*) a zvětšení (*scale*). Bohužel tím snadné věci končí.

S větší námahou zde lze vytvořit kamera a přidat jednotlivé matice a vektory, aby vše fungovalo. Poté vznikne velmi pěkný nástroj pro poznání toho, co jaký vektor v matici ovlivňuje.

Pokud se tedy uživatel dostane přes náročnou přípravu, tak se v tomto programu může velmi dobře naučit nastavovat parametry pro kameru a simulovat různé pohyby s kamerou (přiblížení, švenk).

V programu je připravena řada scén zaměřených na jednotlivé transformace užívané v počítačové grafice, například scéna osvětlující gimbal lock. Bohužel uživatel tuto scénu nenajde někde v menu, aby se k ní rychle dostal a mohl si ve scéně vyzkoušet tuto problematiku. Scéna je velmi pěkně vytvořená. Ke scéně jsou vytvořeny poznámky, které by měly vést k lepšímu pochopení gimbal locku.

Program *I3t* obsahuje několik typů krabiček, které slouží k různým operacím. Jsou zde matice transformací, které se vkládají do sekvencí. Sekvence má 2 vstupy (násobení matice, storage input) a 3 výstupy (násobení matice, model matrix output, storage output) bohužel název těchto vstupů není vidět a zobrazí se až po najetí myši na vstup. U výstupu modelu matice se po najetí myši zobrazí i popis tohoto vstupu. Jednotlivé sekvence nejdou pojmenovávat, ale mají funkci hint do které si můžete připsat vlastní poznámky.

Poté jsou v programu krabičky operátorů, kterých je mnoho druhů, jsou zde operátory pro floaty, vektory, matice, kvaterniony, konverze a pro matice transformací. Program *I3T* obsahuje krabičku pro objekt kameru a pro náhled kamery.

2.4 Chyby v uživatelském rozhraní

Během průzkumu jsem zjistil, že většina studentů používajících program *I3T* nevěděla o připravených scénách. Případně věděla o scénách, ale nevěděla, že jsou myšleny pro výuku. Proto mají autoři programu v plánu přidat do programu funkci tutoriálu, aby bylo jasné, že hlavní funkcí programu je výuka transformací. Je žádoucí, aby se žák k programu vrátil, pokud si nebude vědět při učení/programování rady a zkusil si dané téma, které nechápe, vytvořit v *I3T*.

Připravené scény, které jsou v programu v aktuální verzi jsou uloženy jak soubory na disku a nejsou nějak odlišeny od ostatních scén. Pokud o nich nevíte a nechcete otevřít nějakou scénu, ani na ně nenarazíte. Před připravené scény obsahují připravené objekty a matici k danému tématu. Jsou zde i poznámky s popisem a trochou teorie.

Pro studenta je mnohem lepší začít s prázdnou scénou a po krocích si dané prostředí připravit, lépe poté pochopí danou problematiku. Student by dostával pokyny, co má vytvořit a jak má co nastavit. Scénu by si vytvořil sám a poté by mohl dostat nějaké další úkoly, které by prokázaly, že probírané téma pochopil. Samozřejmě by nemusel začínat úplně od prázdné scény, to by se lišilo tématem. Například u gimbal locku nemá cenu vytvářet scénu, můžete rovnou plnit dané úkoly. Naopak při tvorbě kamerového pohledu je velmi žádoucí, aby si žák vše vytvořil a nastavil sám. Bude se mu to poté velmi hodit, až bude danou věc programovat.

Kapitola 3

Návrh

V této kapitole budou postupně představeny jednotlivé návrhy nového uživatelského rozhraní programu *I3T*. Bude zde představena podoba nových funkcí, především funkce tutoriálu, jejíž přidání vymysleli autoři programu.

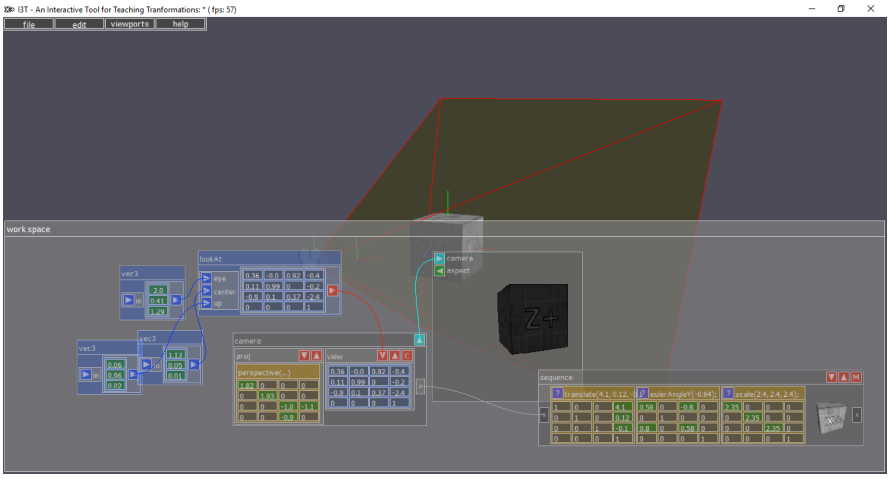
U některých návrhů funkcí bude analyzován vzhled a ovládání v jiných grafických programech, především v programu *Blender*.

3.1 Funkce navrhované studenty

Po analýze nástroje *I3T* víme, které věci se v nástroji nachází. Nyní se podíváme na to, co si žáci přáli, aby se v nástroji nacházelo. Jednotlivé funkce budou v seznamu řazeny podle užitečnosti a náročnosti implementace (nejdůležitější funkce je nejvýše). U některých funkcí bude popsáno, v čem mohou studentům pomoci při výuce nebo jak by měla zhruba vypadat jejich implementace.

- Při vložení kamery by mohla kamera mít již vyplněné matice (viz. obrázek 3.1). Žák nemusí vždy vědět, jaké matice musí kamera obsahovat, aby ukazovala správný pohled. Když uvidí jednotlivé matice, hned lépe pochopí, jak kamera funguje, případně co má v jeho programu/kódu špatně.

- Při vytvoření objektu by k němu mohly být přidány základní matice – posunutí (*translate*), zvětšení (*scale*), případně i rotace (*rotation*), stejně jako jsou přidány ke kostce v úvodní scéně. Matice lze odstranit vždy a je jednodušší je smazat, než postupně jednu po druhé přidávat. Tato možnost by měla jít vypnout v nastavení, protože pokročilejšího uživatele to může rušit, případně by mohli být dvě položky v menu (prázdná sekvence, sekvence s maticemi). Implementace této funkce by neměla být náročná, jen by se při vložení do sekvence vložily i výše zmiňované matice.



Obrázek 3.1: Vložení kamery – automaticky vygenerované matice a náhled na scénu.

- Nastavování barvy objektů v menu, které se objeví při kliknutí pravým tlačítkem na objekt. Menu pro přidání objektu by poté obsahovalo jen jednotlivé objekty bez jejich barevných kombinací.

- Přidání mlhy do scény, případně jen formou checkboxu do viewportu kamery. Do okna s náhledem scény by se přidalo menu, kde by se mohly nastavovat parametry mlhy.

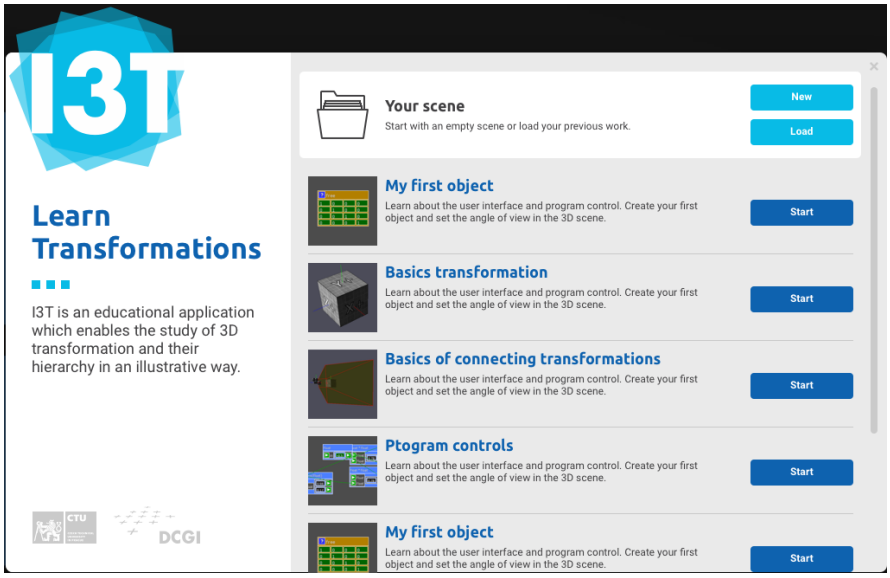
- Základní textury v programu a možnost editace jejich texturovacích souřadnic.

- Různé osvětlovací modely, tím pádem možnost přidání světla a nastavování parametrů pro různá světla. Světla by měli vlastní druh krabičky jako má kamera a parametry by se nastavovaly v této krabičce.

3.2 Grafický návrh z katedry počítačové grafiky

Tyto dva návrhy byly vytvořeny Lukášem Pilkou v roce 2018. Měly nastínit vzhled nové verze programu *I3T*. Bohužel k dispozici mám jen výsledné obrázky a krátký manuál, který obsahuje použité fonty a velikosti jednotlivých nadpisů. Chybí původní zdrojové soubory, v kterých by se daly dělat úpravy. Zcela chybí samostatný obrázek loga *I3T* ve vysoké kvalitě.

Návrh zohledňuje to, že se program má používat hlavně pro výuku. V předešlé verzi sice byly připraveny předpřipravené scény, které měly vždy vysvětlit nějaké téma, ale většina uživatelů o nich nevěděla. Studenti, kteří je objevili, se v nich velmi těžko orientovali. V tomto návrhu viz Obr. 3.2 na tu-



Obrázek 3.2: Návrh úvodní obrazovka od Lukáše Pilky s výběrem jednotlivých lekcí, případně s volbou vytvoření/otevření vlastní scény. Obrázek byl převzat z [16]

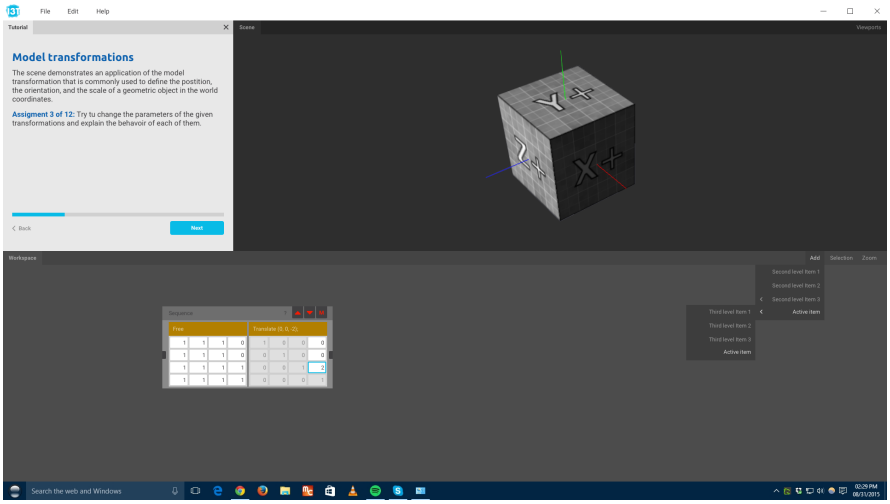
toriály (výukové lekce) uživatel narazí hned po otevření programu na úvodní obrazovce.

Výuková lekce viz Obr. 3.3 provede žáka krok po kroku daným tématem. Tento způsob se jeví jako velmi úspěšný. Ze svých osobních zkušeností vím, že učení se po krocích člověka o mnoho více motivuje a hned danou činnost nevzdá. Když má uživatel jen počáteční scénu a daný cíl, tak může daný úkol lehce vzdát, protože se zasekne a nebude se mu chtít řešit, kde udělal chybu.

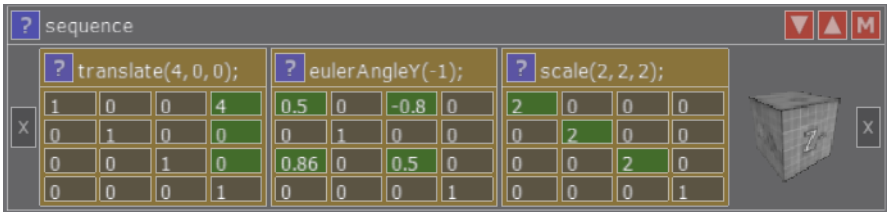
3.3 Design systému krabiček

V této sekci se budeme věnovat úpravě vzhledu jednotlivých krabiček. Podobu sekvence v aktuální verzi můžete vidět na obrázku 3.4. Nejdůležitější bude změna umístění jednotlivých vstupů a výstupů a jejich názvů, aby byly lépe pochopitelné.

S názvy vstupů a výstupů v programu mají uživatelé v aktuální verzi programu velké problémy. Nechápu k čemu jednotlivé vstupy a výstupy slouží. Názvy vstupů a výstupů se zobrazují až po najetí myší na výstup/vstup, výjimkou jsou vstupy a výstupy pro násobení, kde se nezobrazuje ani název. U výstupu s názvem *Model Matrix Output* je i věta popisující funkcionalitu tohoto vstupu. Zbylé dva vstupy se jmenují *storage input/output* z čehož uživatel nemůže poznat k čemu slouží. I umístění vstupů a výstupů je velmi nestandardní.



Obrázek 3.3: Návrh hlavní obrazovky od Lukáše Pilky z katedry počítačové grafiky. Je zde přidané okénko s tutoriálem a jednotlivými kroky k dokončení úkolu. Obrázek byl převzat z [17]



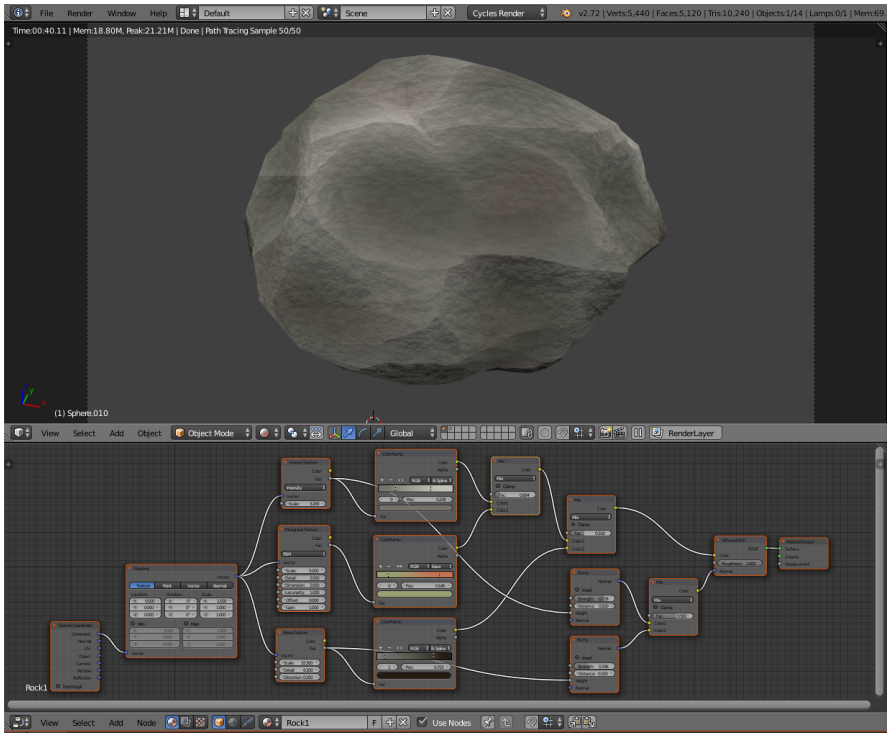
Obrázek 3.4: Vzhled krabičky v aktuální verzi programu

3.3.1 Existující řešení systému krabiček

Podobný systém propojování krabiček, který používá *I3T* používá grafický program *Blender*, který tento systém nazývá *Nodes* [18] a pomocí něho řeší hned několik věcí.

Nodes se používá v nástroji na vytváření materiálů u *Cycles render* viz Obr. 3.5. Systém zde je velmi intuitivní a dají se s ním vytvářet velmi kvalitní materiály. Dále *Blender* využívá *Nodes* při postprodukci výstupů z renderu viz Obr. 3.7. Jelikož se z programu *Blender* postupem času stává multifunkční program umožňující i úpravu obrázků u které používá již ověřený systém *Nodes*, jenž uživatelé již dobře znají a dobře se jim v něm pracuje.

Systém *Nodes* je uživatelsky přívětivý. V *I3T* jsou některé vstupy a výstupy popsány v horní liště krabičky, aby ušetřily místo. V *Nodes* je každý vstup i výstup pojmenován a označen tečkou různé barvy, podle druhu vstupu/výstupu viz Obr. 3.6.



Obrázek 3.5: Editor materiálu *Nodes* v programu *Blender Cycles*. Obrázek byl převzat z [19]

Jednotlivé uzly se dají dávat i do skupin[2], což zlepšuje přehlednost. Celkově člověk při prvním pohledu zjistí, co daný uzel (*node*) dělá a co je jeho vstupem a výstupem. *I3T* také odlišuje barvy vstupů/výstupů, ale ty většinou korespondují s barvou krabičky, takže si to uživatel moc neuvědomuje.

Navzdory tomu, že obsahy uzlů se liší, je celý systém stále přehledný. Obsahy jsou jen různě veliké, ale na tomto příkladu je vidět, že zde můžeme zobrazovat křivky, náhledy obrázků nebo nastavování parametrů a systém zůstává stále přehledný.

Pro začínajícího uživatele je těžší se zorientovat v nabídce uzlů. *Blender* používá klávesové zkratky pro nejčastěji používané uzly a pro otevření nabídky jejich výběru. Pokročilý uživatel díky tomu může ušetřit mnoho času, pokud zná a umí klávesové zkratky.

[2]Ukázka práce se skupinami v programu *Blender*. `https://code.blender.org/2012/01/improving-node-group-interface-editing/`

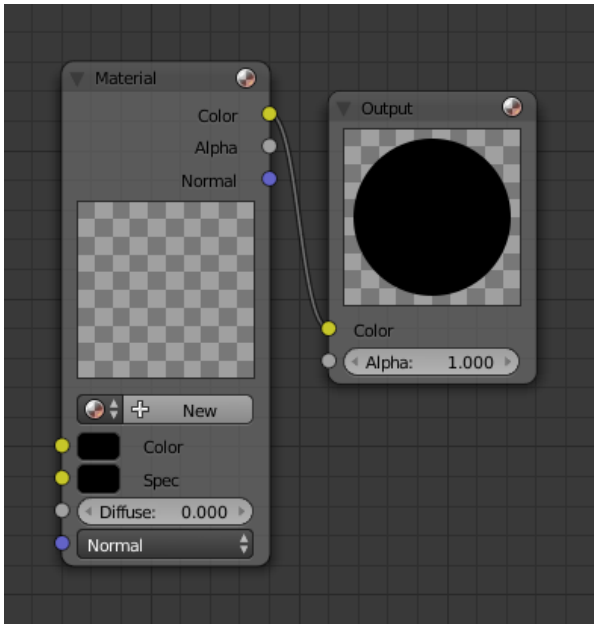
3.3.2 Srovnání *Nodes* s řešením v *I3T*

Pro shrnutí rozdílů mezi programy *I3T* a *Blender* my sepsal vedoucí práce pan Felkel několik otázek na funkce programu, které ho zajímali a já jsem na ně odpověděl.

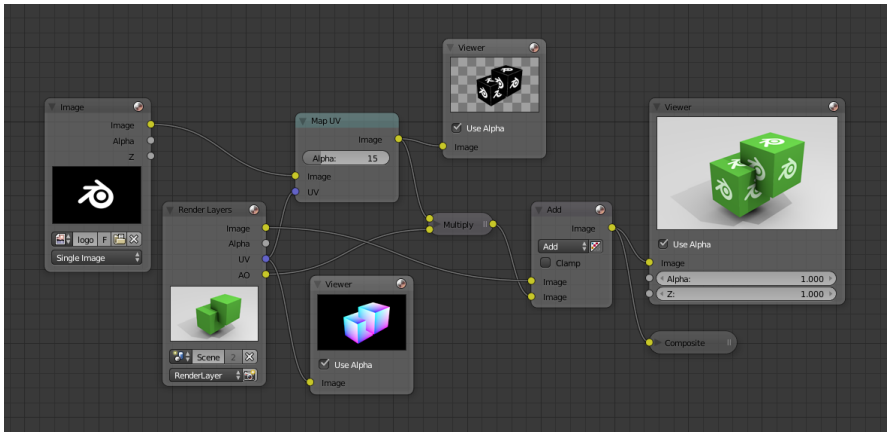
- Jak se přidá modul? Jak se liší způsob přidání modulu/krabičky do GUI u *I3T* a zkoumaného editoru?

  Blender: První možnost přidání modulu je v položce přidat (*add*), kde jsou další volby a v nich teprve finální krabičky. Další možností je zkratka Shift + A která otevře rovnou nabídku, která je uvnitř menu přidat (*add*). Navíc můžete hned začít psát název daného modulu a vyhledat ho. Poslední možnost je vyhledat daný uzel (*node*) pomocí mezerníku, který v programu *Blender* funguje pro vyhledávání funkcí. Zde použijeme funkci vyhledat a přidat (*search and add node*) viz Obr. 3.8, což nám vyvolá menu s jednotlivými krabičkami, v kterém můžeme vyhledávat (toto vyhledávání využije hlavně pokročilý uživatel, který zná názvy modulů).

  I3T: V *I3T* nefungují žádné klávesové zkratky, které by vyvolaly menu. V tomto případě to asi moc nevadí, protože se menu pro přidání vyvolá kliknutím pravým tlačítkem. Věc, která mi v *I3T* vadí, je že se krabička



Obrázek 3.6: Základní nastavení *Nodes* pro materiál. Všimněte si, že výstupy mají různé barvy, které jsou ale u všech uzlů stejného typu neměnné. Obrázek byl převzat z [20]



Obrázek 3.7: Použití *Nodes* pro postprodukci v programu *Blender*. Obrázek byl převzat z [21]

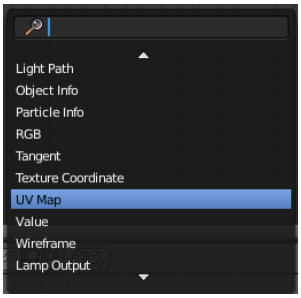
po kliknutí nevybere (pro vybrání slouží zkratka Shift + klik levým tlačítkem). Což většinu uživatelů zmate a nedokáží ani smazat krabičku, protože jsou zvyklí vybírat pomocí kliku levým tlačítkem, případně Ctrl + klik levým tlačítkem a poté použít zkratku pro kopírování nebo pro smazání.

- Lze krabičky a části zapojení jednoduše duplikovat?

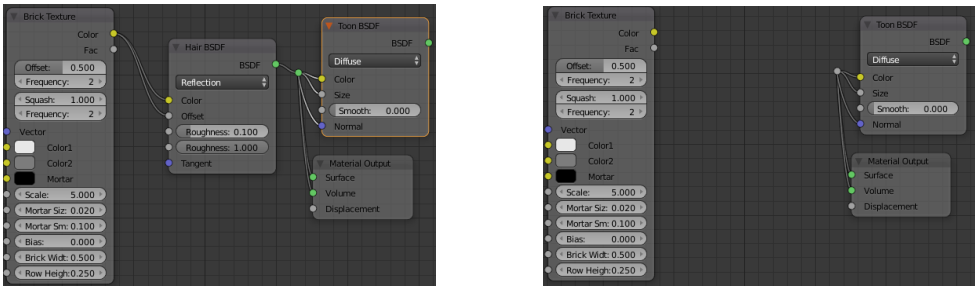
  I3T: Ano, v *I3T* lze objekt duplikovat pomocí vybrání + Ctrl a kliknutí levým tlačítkem myši

  Blender: Ano, v programu *Blender* si vyberete co chcete duplikovat a zmáčknete Ctrl + D .

- Hodila by se opačná možnost smazat krabičku v grafu tak, aby se místo ní propojili sousedi drátem? A-B-C nahradit A-C, pokud to má smysl...



Obrázek 3.8: Menu jednotlivých modulů, ve kterém jde vyhledávat a nebo ho procházet šipkami. Velmi zrychluje práci v *Nodes*



Obrázek 3.9: Ukázka rozbočovače, který zachová rozvětvení vstupů. Lze přidat pomocí zkratky shift + levé tlačítko a tažení přes spoj.

V *I3T* ani v základním *Blenderu* to nejde (nenašel jsem tuto možnost), ale pro *Blender* existuje rozšíření, které toto umožňuje a celkově zlepší práci s *Nodes* – Node Wrangler [22].

- Jak se vloží modul mezi dva jiné – lze jej přímo položit na spojovací drát a on se propojí? Nebo se ručně drát přepojí na vstup nového modulu a výstup nového modulu na další v řadě?

  I3T: Bohužel *I3T* tuto funkci nemá, ale byla by to velmi užitečná nová funkce. V *I3T* tuto funkci může nahradit například krabička pro float, přes kterou vše zapojíme a při smazání sekvenci zůstanou spojení zachovány.

  Blender: Ano, v *Blenderu* stačí dát modul na spojovací drát a on se sám přepojí.

- Někdy je na výstup krabičky A zapojeno třeba 5 dalších. Dala by se vymyslet funkce, která na výstup A dá kopírovací modul, do kterého bude zapojeno těch 5 ostatních a jeho vstup se zapojí na výstup A? Pak půjde jednoduše vyměnit A za jinou bez nutnosti propojovat znova jejích 5 výstupů.

  I3T: bohužel tuto funkci neobsahuje, ale byla by to velmi užitečná nová funkce programu *I3T*.

  Blender: na tohle má něco jako rozbočovač, který vytvoříte, když držíte Shift + kliknete levým tlačítkem a přejedete přes spoj. Tímto se vytvoří uzel nezávislý na krabičce který zůstane i při odstranění krabičky se vstupem viz Obr. 3.9.

- Jak se propojují dráty? Klik na výstupu a poté klik na vstupu? Nebo klik na jednom a táhnutí na druhý (je vidět natahovaný vodič během interakce?)?

  I3T: Zde je to klik a táhnutí (*click and drag*) na druhý vstup, vodič má barva stejnou jako zapojovaný vstup/výstup.

Blender: Zde je to klik a táhnutí (*click and drag*) na druhý vstup, vodič je viditelný a žlutý, aby se odlišil od ostatních šedě zbarvených vodičů.

- Rozsvícení (*highlighting*) možných vstupů: Když kliknu při sestavování mapy na výstup modulu, rozsvítí se mi možné vstupy dalších krabiček, kam bych mohl drát zapojit?

  I3T: Nerozsvěcuje jednotlivé vstupy, ale má datové typy barevné a neumožňuje propojit odlišné typy.

  Blender: Nerozsvítí, ale *Blender* umožňuje dát jakýkoliv výstup do vstupu, zároveň však barevně odlišuje jednotlivé druhy výstupů a vstupů.

- Lze přidat jednoduše generátor vstupní hodnoty?

  I3T: Zde kliknu pravým tlačítkem na float vstup operátoru a vyrobí se krabička na float. Pole, na které jsem klikl přestane být editovatelné.

  Blender: Toto má *Blender* vyřešené velmi zajímavě. Pokud lze generovat vstup, který je jednoduchý, je zde rovnou pole, které můžete vyplnit, pokud tam však pošlete vstup, tato volba zmizí.

- Jak se editor brání vytvoření cyklů v grafu?

  I3T: neumožní vytvořit cyklus.

  Blender: cyklům nebrání.

- Jak se vytvářejí bloky modulů? Které vstupy a výstupy modulů vložených do bloku se ukáží jako vstupy a výstupy na bloku? Je to automatické nebo se ručně vyberou např. při vytváření bloku?

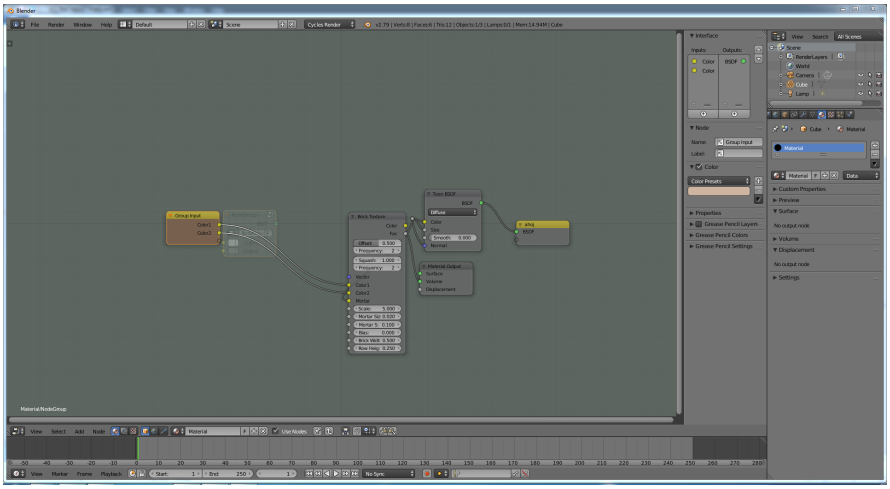
  I3T: Bloky modulů v programu nejdou vytvořit. Lze zde shlukovat jednotlivé matice do sekvencí. Toto řešení, ale nedokáže ušetřit místo, protože sekvence se nedají nijak zmenšit.

  Blender: Blok modulů se vytvoří, tak že vyberete požadované moduly a zmáčknete ctrl + G, vstupy pro blok a výstupy se nastavují ručně pomocí propojování drátů. Toto nastavení lze kdykoli změnit v editoru viz Obr. 3.10, do kterého se dostaneme po zmáčknutí tab, když máme danou skupinu vytvořenou. Tyto skupiny velmi zpřehledňují práci v *Nodes*.

- Je tam nějaká forma automatického rozmístění bloků (automatický layout)?

  I3T: obsahuje jen vložení do mřížky (po stisku klávesy Alt).

  Blender: umožňuje jen zvětšení (*scale*), které lze použít jen pro osu x nebo y, pomocí zmáčknutí klávesy S + X/Y. Automatické uspořádání umožňují pluginy Node wrangler [22] a NodeArrange [23]. Unreal Engine [24] toto zarovnání umožňuje v základu. Má i několik možností na výběr. [25]



Obrázek 3.10: Ukázka editoru pro skupinu, vpravo nahoře lze nastavovat další detaily toho, jak vypadají vstupy a výstupy a přidávat názvy pro přehlednost.

3.4 Mé návrhy

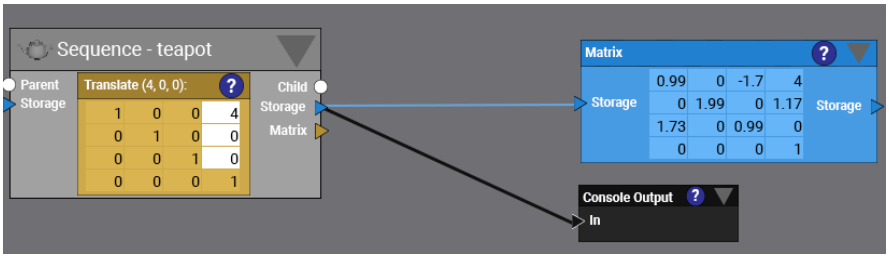
V následují kapitole budou popsány všechny mnou vytvořené či převzaté návrhy uživatelského rozhraní. Budou zde popsány nové funkce (tutoriál, okno konzole, přiblížení matice a minimalizace matic) a jejich vzhled. Některé ze starších funkcí projdou grafickými úpravami (krabičky, menu, atd.). Bude změněno rozmístění jednotlivých menu a celý program bude navrhován jako systém jednotlivých oken, jejichž rozložení půjde měnit.

3.4.1 Systém krabiček

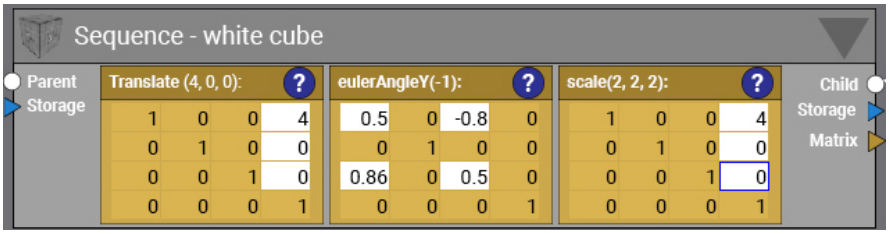
V této sekci budou mnou navržené formáty krabiček. Krabičky v aktuální verzi mají hned několik problémů. Uživatel na první pohled z designu krabičky nepozná, jaké jsou její vstupy a výstupy, toto hlavně platí pro krabičku sekvence.

Aktuálně jsou u krabičky sekvence nahoře umístěny dva trojúhelníky pro vstup a výstup hodnot z aktuální sekvence a pak znak M, který předá matici obsahující součin všech předešlých matic. Dále jsou zde výstupy pro spojování označené křížkem. Vzhled sekvence byl ukázán již v úvodu této kapitoly na obrázku 3.4. Toto značení nový uživatel jen velmi těžce pochopí. Jednotlivé sekvence se nedají pojmenovávat ani nijak zmenšit, pokud tedy máme nějakou složitější scénu, je problém s orientací ve scéně. Lze však zmenšit celou scénu, ale pak jsou krabičky velmi malé, nejsou vidět jejich hodnoty a špatně se editují.

Dráty spojující jednotlivé krabičky, budou mít vždy barvu vstupu, do kterého vedou viz Obr. 3.11. Spojování bude fungovat jak pomocí tažení (*drag*



Obrázek 3.11: Ukázka propojení sekvence s operátorem a konzolou



Obrázek 3.12: Návrhy se stejně velkými vstupy, jednotlivé vstupy jsou barevně označeny a popsány.

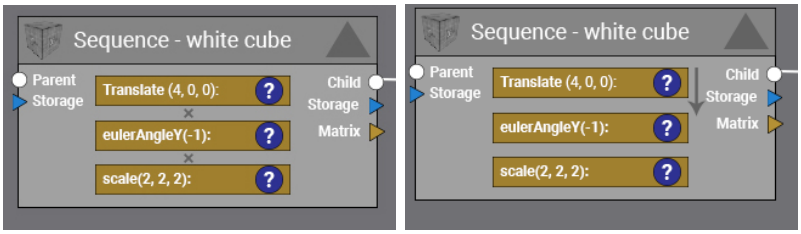
and drop), tak i pomocí kliku na první vstup/výstup a poté na druhý. Při kliku vstup změní barvu, aby bylo vidět, že je vybraný.

3.4.1.1 Návrh jedna – výstupy stejně veliké

Pro větší přehlednost jsou vstupy a výstupy sekvence přesunuty na strany a přidány jim popisky. Jednotlivé vstupy a výstupy jsou barevně i tvarem odlišeny. Pro násobení je použit symbol bílého kolečka, pro výstup matice uvnitř sekvence modrý trojúhelník. Modrou barvu jsem zvolil proto, že jednotlivé výstupy se často zobrazují pomocí operátorů, které mají modrou barvu. Oranžový trojúhelník slouží pro výstup násobení všech sekvencí. Oranžovou barvu jsem zvolil proto, že je používána i pro matice transformací, tedy ty matice, které jsou umístěny uvnitř sekvence.

V maticích jsem buňky, které jsou zamknuté, vybarvil světle oranžově a aktivní pole bílou barvou. Symbol pro nápovědu k matici jsem dal do modrého kolečka. Otázkou je, zda se tento symbol pro něco hodí. V aktuální verzi lze pomocí něj přidat nápovědu k matici. Po zavedení tutoriálu by se už nemusel téměř používat. Mohou ho však využít tvůrci nových scén pro své poznámky, ale to by se mohl přesunout do hlaviček sekvence.

Dále jsem pro zpřehlednění přidal název sekvence podle toho, co obsahuje a možnost přejmenování sekvence. Změna názvu matice nebude umožněna, jedinou výjimku tvoří *free matice*, u které by bylo dobré si popsat co dělá. Miniaturu jsem přesunul zleva do horního rohu, kde tvoří něco jako logo dané



Obrázek 3.13: Minimalizace sekvence. V dolních obrázcích jsou dva způsoby pro zvýraznění násobení matic.

sekvence. Do pravého horního rohu jsem umístil trojúhelník, který slouží pro zmenšení okna viz Obr. 3.12.

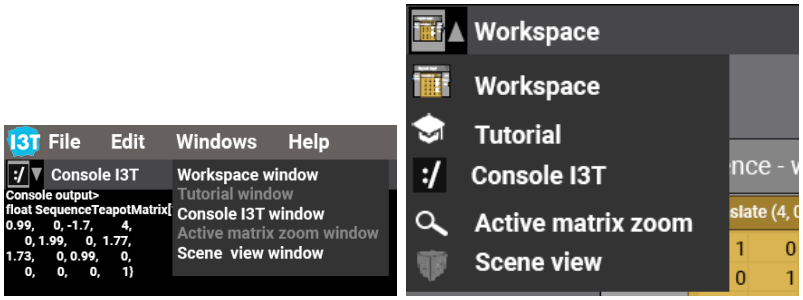
Jak je vidět na obrázku 3.13 minimalizací matice se dá ušetřit hodně místa, ale je tu problém s pochopením pořadí jednotlivých matic. Proto jsem navrhl dvě řešení, jedno za pomoci šipky vlevo, která znázorňuje směr násobení, další přidáním křížků viz Obr. 3.13. Způsob zdůraznění násobení jsem otestoval v kapitole pluralistické testování 4.4.1 a lepší variantu jsem použil ve finálním návrhu. Dále bude potřeba otestovat jestli nevyměnit symbol pro maximalizaci a minimalizaci.

3.4.2 Systém oken

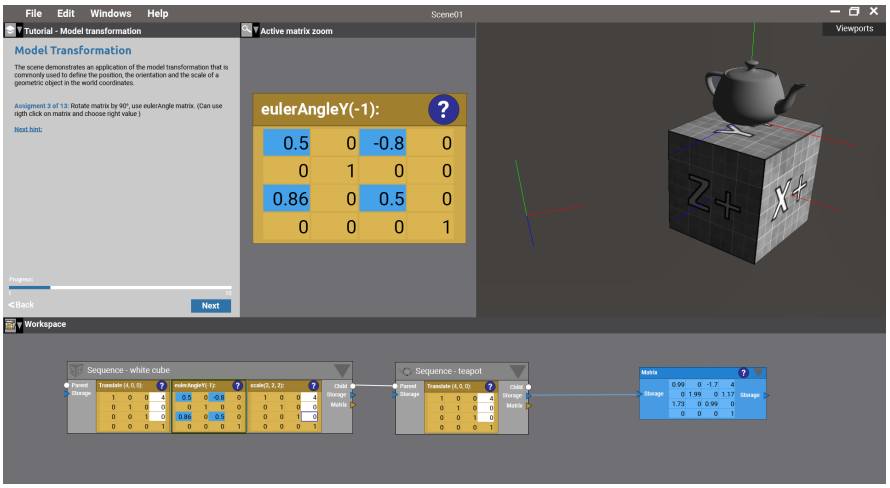
Program *I3T* má problém s velikostí pracovní plochy, která v aktuální verzi jde jen roztahovat nahoru a dolů. V novém návrhu je v programu vytvořen systém jednotlivých oken, který používá například *Blender*. Uživatel si tak bude moci sám nastavit rozložení programu, případně, pokud používá více monitorů, tak si bude moci rozdělit program do několika oken a přesunout okno s krabičkami na druhý monitor a zvětšit ho. Typ okna by se určoval v jeho levém rohu viz Obr. 3.14.

Navrhuji do programu přidat hned několik funkcionalit, každá bude mít i své okno. Hlavní okna, která jsou již v aktuální verzi, jsou okna s pracovní plochou a náhledem scény. Dále bude přidáno okno pro tutoriál, pro zvětšení aktuální matice a pro konzoly. Stávající okna budou upravena a budou přidány nové možnosti, například v pracovní ploše lupa, která zobrazí všechny krabičky. V některých programech se používá podobné záchranné tlačítko, například domeček v programu *Blender*.

V horním panelu přibylo menu oken (*windows*) pomocí kterého si bude uživatel moc přidat jednotlivá okna do scény. Bude to jen jedna z možností. Další možnost bude systém, který okno rozpůlí (okna se rozdělí na dvě, podobně si můžete okna přizpůsobit v programu *Blender*) a poté se bude moc změnit režim okna na jiné. V menu budou nepoužívaná okna tmavší, když si ho vyberete, budete si moci zvolit, kam ho umístit.



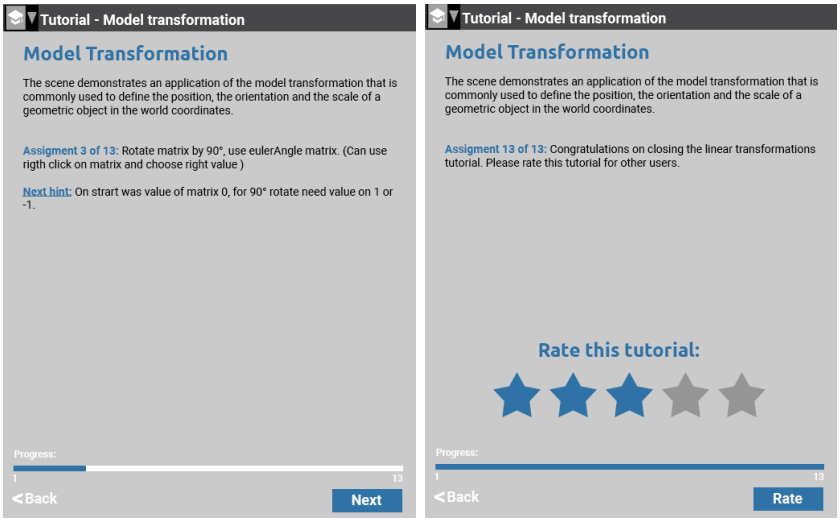
Obrázek 3.14: Výběr druhu okna pomocí menu oken (*window*) a menu v levém horním rohu pracovní plochy (*workspace*)



Obrázek 3.15: Kombinace okna tutoriálu a přiblížení matice. Je zde zobrazena matice eulerova úhlu, která je vybrána v pracovní ploše. Zvýrazněné hodnoty slouží jako nápověda k tutoriálu

3.4.2.1 Okno na přiblížení matice

Program používá při výuce na přednáškách a cvičeních. Proto chtěli učitelé funkcionalitu pro jednoduché zvětšení aktuální matice, aby mohli vysvětlit jednotlivé parametry. Aktuální verze obsahuje sice klávesovou zkratku D , která přiblíží aktuálně vybranou matici, ale poté není vidět zbytek matic a studenti ztratí kontext jaké další matice jsou ve scéně použity. Toto okno je velmi jednoduché, otevře se v něm aktuálně vybraná matice přes celou šíři okna. To se dá velmi dobře použít při plnění tutoriálu viz Obr. 3.15, protože se v něm otevře matice, do které se mají doplnit hodnoty. Pokud však vyberete jinou matici, bude v okně pro přiblížení ta aktuálně vybraná. Matice do které se mají doplnit hodnoty dle instrukcí tutoriálu se v okně pro přiblížení objeví znovu, když kliknete mimo a zrušíte tak označení aktuálně vybrané matice.



Obrázek 3.16: Vlevo obrázek s nápovědou pro tutoriál a vpravo je návrh hodnocení tutoriálu.

3.4.2.2 Okno s tutoriálem

V aktuální verzi jsou předpřipravené scény, při otevření se zobrazí i poznámky s popisem scény. Bohužel žákům tyto poznámky moc nepomohly, proto jsem navrhl přidání okna s tutoriálem. V tomto okně budou postupně popisovány jednotlivé kroky, které je potřeba udělat, a teorie ohledně probírané látky. Jednotlivé tutoriály budou moci tvořit i uživatelé a rozšiřovat tak obsah programu. Nyní mohou v programu *I3T* připravit scénu s popisem problému a teoretickým vysvětlením, kterou mohou sdílet s ostatními uživateli.

Kvalitu jednotlivých tutoriálu budou uživatelé hodnotit po jeho dokončení pomocí systému hvězdiček (1 - 5) viz Obr. 3.16. Bude potřeba otestovat, zda změna tlačítka další (*next*) na ohodnotit (*rate*) nebude uživatele mást, nebo jestli by se tlačítko nemělo přemístit někam jinam, případně že by se hodnocení odeslalo hned po výběru hvězdiček. Dokud uživatel nevybere počet hvězdiček budou šedé.

Při plněný jednotlivých úkolů bude program uživateli napovídat zvýrazněním jednotlivých políček, do kterých má uživatel zadávat hodnoty. Pokud bude mít uživatel otevřeno i okno pro přiblížení matice, zobrazí se v něm matice, ve které se mají změnit hodnoty viz Obr. 3.15.

3.4.2.3 Okno konzole

V aktuální verzi lze v nastavení zapnout konzolu. Jsou zde jen vypisovány hlášky z programu, pro uživatele zde není nic důležitého. Konzole se otevře v novém okně. Při průzkumu jsem zjistil, že studenti by chtěli funkci, pomocí

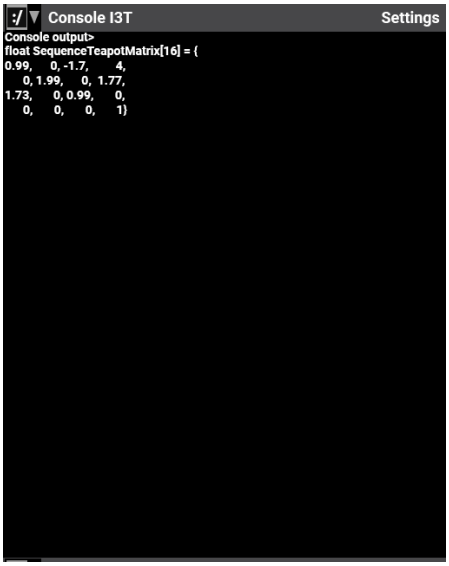
které by mohli jednotlivé matice vybudované ve scéně převést do nějaké formy, která se dá přímo použít v programu.

V mém návrhu viz Obr. 3.17 je konzole jeden druh okna a lze do ní z pracovní plochy za pomoci krabičky konzole přidat výstup z programu. Krabička konzole je velmi jednoduchá, má jen jeden vstup, do kterého můžete přivést storage nebo maticový výstup. Po připojení do krabičky se v konzoli objeví vygenerovaná C++ matice, kterou si uživatel může jednoduše zkopírovat.

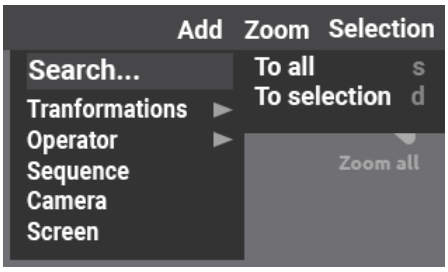
3.4.2.4 Úprava okna s pracovní plochou *(Workspace)*

Okno s pracovní plochou viz Obr. 3.18 bude doplněno o menu, které se bude nacházet v pravém horním rohu, podobně jako v návrhu od Lukáše Pilky. Bude rozděleno do tří kategorií a to pro přidání, výběr a přiblížení. V aktuální verzi jsou všechny tyto funkce dostupné pod jedním menu po kliknutí pravým tlačítkem.

Menu přidat (*add*) půjde také spustit pravým tlačítkem myši a bude navíc obsahovat možnost vyhledávání pro rychlejší nalezení potřebné krabičky. Tato funkce by měla vylepšit ovladatelnost programu, protože nyní mnoho uživatelů ví, co hledá, ale neví, v jakém podmenu se tato funkce nachází.



Obrázek 3.17: Okno konzole s výpisem hodnot



Obrázek 3.18: Přidané menu do pravého horního rohu pracovní plochy

Kapitola 4

Testování

Testování je velmi důležitou součástí vývoje software. Odhalení chyb v návrhu před jejich implementací ušetří vývojářům mnoho času. Software se testuje během celé doby jeho vývoje. Pro každé stádium vývoje jsou popsány jiné druhy testování.

Krug ve své knize *Web design - nenuťte uživatele přemýšlet!* [26] říká, že stačí provádět mnoho menších testování s třemi až čtyřmi testery, kteří odhalí většinu chyb, protože zásadní chyby se u jednotlivých testerů opakují. Dále tvrdí, že skupinové testy neslouží úplně pro testování použitelnosti programu. Jsou spíše určeny pro zjištění, jestli daná podoba dává uživatelům smysl a poznají podle názvu funkce, co dělá. Proto se používá hlavně při testování návrhu designu aplikace. [26]

Testování použitelnosti popisované panem Krugem je velmi užitečné pro tuto studentskou práci, protože tvrdí, že není důležité vybírat uživatele na základě nějakých předem daných vlastností. Hlavní je mít nějaké uživatele na kterých můžete program otestovat a nejlépe co nejčastěji. To proto, že testování s jedním člověkem na začátku a v průběhu vývoje je lepší, než testování s 50 uživateli těsně před vydáním programu. [26]

Nejdříve je potřeba otestovat grafické návrhy vzniklé v předchozí kapitole, pro které se nejlépe hodí jedna z forem skupinového testování. Na základě tohoto testování vytvořím prototyp a budu se zabývat testováním použitelnosti mého návrhu.

4.1 Způsoby testování použitelnosti

Nejprve představím několik často používaných metod testování použitelnosti, kdy a jak se používají a jaké mají výhody a nevýhody.

Pluralistický průchod se používá při začátcích vývoje aplikace. Při testování se používají obrázky představující program, které mohou být vytištěny nebo zobrazovány na monitoru. Testování se účastní skupina pěti

až osmi respondentů, vývojový tým, moderátor a další osoby podílející se na vývoji aplikace. Pro účely testování je sestaven scénář, který respondenti postupně procházejí. Po jednotlivých krocích probíhá diskuze mezi vývojovým týmem a respondenty. Z ní vývojový tým zjistí, co vše je potřeba upravit. [27]

Heuristická analýza spočívá v hledání porušení předem daných pravidel (heuristik) jedním nebo více odborníky. Existuje mnoho různých pravidel na základě, kterých se může daná aplikace hodnotit. Experti se vžívají do předem vytvořených osobností (person), které jsou vytvořeny pro každou skupinu uživatelů, kteří budou aplikaci používat. [28]

Nejznámějšími heuristickými pravidly použitelnosti je deset bodů použitelnosti podle Jacoba Nielsena:

1. Viditelnost stavu systému
2. Propojení systému a reálného světa
3. Uživatelská kontrola a svoboda
4. Konzistence a standardizace
5. Prevence chyb
6. Rozpoznání místo vzpomínání
7. Flexibilní a efektivní použití
8. Estetický a minimalistický design
9. Pomoci uživatelům pochopit, poznat a vzpamatovat se z chyb
10. Nápověda a návody [27]

Kognitivní průchod se používá při rané fázi testování aplikace. Aplikaci netestují její reální uživatelé, ale testeři (experti). Výsledky kognitivního průchodu tedy mohou být zkreslené. K testování je potřeba prototyp scény a připravený scénář, podle kterého testeři postupují. I u tohoto typu testování je žádoucí, aby byli pokryty všechny skupiny uživatelů, proto experti využívají persony, podle kterých se při průchodu rozhodují. Kognitivní průchod nemůže nahradit testování s uživateli.

Uživatelské testování se provádí v testovací laboratoři. Testovaný uživatel je v jedné místnosti spolu s moderátorem, který ho seznámí s tím, proč zde je a co se bude dále odehrávat. Důležité je zmínit, že netestujeme uživatele, ale aplikaci. Testovaný je požádán, aby říkal své myšlenky nahlas. Ve vedlejší místnosti sedí vývojový tým, který sleduje průběh testu a píše si poznámky. K sledování se používají kamery, které zabírají uživatele a jejich obraz je společně s obrazem obrazovky přenášen do místnosti, kde se nachází vývojový tým. Některé laboratoře místo kamery používají polopropustné zrcadlo. Testovaného uživatele nesmí

> nic rušit a musí být co nejvíce v klidu. Po skončení testován uživatele proběhne diskuze mezi uživatelem a vývojovým týmem. Programátoři opraví nejzásadnější chyby zjištěné během testování, poté může začít další kolo testování. Pro jedno kolo testování stačí 3–4 účastníci. [26]

Další podrobnosti a nápovědy ohledně výše zmiňovaných způsobů testování, ale i dalších možností můžete najít na stránce `https://www.usabilitybok.org/`. Stránka obsahuje popis jednotlivých testování, rady jak jednotlivé testy provést a jak se na ně připravit, ale i teoretický popis a odkazy na literaturu zabývající se daným druhem testovaní. Stránku jsem využil hlavně při přípravě jednotlivých testů pro kontrolu jednotlivých bodů, které má test obsahovat a jestli mám vše připraveno. [29]

4.2 Skupiny uživatelů

Program budou používat čtyři skupiny uživatelů. První skupinou jsou lektoři, kteří budou tento program používat při výuce. Druhou skupinou jsou studenti předmětu, kteří se v tomto programu učí transformace, tuto skupinu jsem rozdělil na dva případy, protože program používají studenti během výuky. Dále ho mohou využívat k učení na zkoušku, případně k tvorbě tutoriálů (v aktuální verzi k přípravě scén pro ostatní uživatele programu). Tyto dvě skupiny se poměrně dost liší. Třetí skupinou jsou „programátoři“, vytvářející jednotlivé scény a tutoriály. Čtvrtou skupinou uživatelů budou nadšenci do počítačové grafiky, využívající program pro tvorbu tutoriálu i pro simulace. Tato skupina je asi nejobecnější a měla by být největší.

Jednotlivé skupiny budou představeny podle odhadované velikosti jednotlivých skupin od nejmenší (nejspecifičtější) po největší (nejobecnější).

4.2.1 Učitelé

Učitelé používají program na přednáškách a cvičeních pro vysvětlení jednotlivých transformací a jejich skládání. Popisují zde hodnoty jednotlivých matic a jaký vliv mají na scénu. Chtějí mít k dispozici nějaký nástroj pro jednoduché zvětšení, případně zvýraznění nějaké matice, aby byla přehledná s dobře viditelnými konkrétními hodnotami.

Na přednášce se nebudou ukazovat nějaké složitější scény, protože by studenti nepochopili, o co ve scéně jde. Učivo budou učitelé ukazovat na jednoduchých příkladech.

Dále budou vyrábět ukázkové scény ilustrující probíranou látku. Jelikož se program používá i na cvičeních, musí se připravit i scény pro cvičení, případně nějaké úkoly na doma. Velmi výjimečně bude učitel vyrábět tutoriál.

Persona: Ing. Oliver Vrba je 50letý učitel na ČVUT. Vystudoval FEL ČVUT, poté několik let pracoval, jako vývojář nejrůznějších programů pro

práci s 3D prostorem. Osmým rokem učí na ČVUT základy počítačové grafiky a snaží se předmět maximálně oživit a přiblížit studentům. Například využíváním nejnovějších technologií.

Podle Olivera mají mít studenti více možností, jak se dané téma naučit – pomocí webových stránek, video tutoriálů, stránek jako je *Geogebra*, případně pomocí našeho programu. Oliver rád zkoumá nové programy a výzkumy zabývající se počítačovou grafikou. Svůj život si bez moderních technologií nedokáže téměř představit.

Typické použití programu: Otevření předem připravené scény na přednášce. Zapnutí zoom matice. Oliver postupně mění parametry a vysvětluje studentům, co právě udělal. Do scény přidá novou sekvenci a zeptá se, jakou tam má přidat matici, aby mohl s daným objektem pohybovat.

4.2.2 Studenti - samostudium

Studenti budou nejvíce dbát na přehlednost scény a kvalitu jednotlivých tutoriálů. Chtějí používat jim známé zkratky z ostatních programů a lehce se orientovat v programu. Jednotlivé kroky tutoriálu by jim měly, co nejlépe popsat, co se odehrává a čeho změnou jednotlivých parametrů docílí.

Důležitou částí programu pro ně je tutoriál. Pro učení jim stačí malé scény, nebudou vytvářet složité scény. Chtějí si zde dané téma vyzkoušet a poté své zkušenosti uplatnit v programování. Dále by chtěli vytvořené matice v programu vygenerovat do C++ reprezentace matice, které by mohli použít ve svém kódu.

Pokročilejší uživatelé budou tvořit tutoriály pro své spolužáky, aby jim na nich mohli zábavně vysvětlit probíranou látku.

Persona: Pavel Novák je 21letý student druhého ročníků FIT ČVUT v Praze oboru počítačová grafika. Předtím vystudoval gymnázium v Liberci. Jeho koníčkem je fotografování a natáčení videí. Orientuje se v grafických programech jako jsou například *Adobe Photoshop*, *Adobe Premiere* a *Blender*.

Tento rok absolvoval předmět BI-MGA (Multimediální a grafické aplikace). Baví ho poznávat nové programy a hrát graficky zajímavé hry na telefonu. Proto je pro něj důležité zajímavé GUI (grafické uživatelské rozhraní) a jednoduchost ovládání. Právě studuje předmět počítačová grafika a zjistil, že v něm má značné mezery. Moc se na cvičení nepřipravuje, proto nechápe, jak má doplňovat jednotlivé parametry do kódů na cvičení. Program používal na cvičení a rozhodl se ho využít ke studiu ke zkoušce.

Typické použití programu: Student zjistil, že neví jak přesně fungují transformace kamery a vzpomene si na cvičení, kde jim učitel říkal o tutoriálu na toto téma. Otevře tedy *I3T* a najde nejlépe hodnocený tutoriál na toto téma. Poté postupuje po jednotlivých krocích a čte informace ohledně teorie. Nakonec si vytvořené matice nechá vypsat do konzole, aby je mohl použít ve svém programu.

4.2.3 Studenti na cvičení

Tyto studenti používají program na cvičení a většinou se řídí pokyny cvičícího. V případě, že nebudou něco vědět, mohou se zeptat vyučujícího. Většinou budou pracovat na nějaké zkušební scéně, případně budou postupovat podle tutoriálu, kde bude vše popsáno.

Budou očekávat podobné ovládání, jako mají ostatní 3D programy, se kterými se setkali. Většina z nich v minulém semestru pracovala v programu *Blender*. Budou se postupně učit jednotlivé funkce, které pak mohou využít pro tvorbu své semestrální práce, případně k učení na zkoušku.

Persona: Petr Jirmásek je 20letý student FIT ČVUT. Pochází z Karlových Varů, kde vystudoval gymnázium. Rád hraje počítačové hry, jeho neoblíbenější hra je *World of Warcraft*. Fotografuje a navštěvuje různé kulturní události. Právě začal studovat 4. semestr a čeká ho tedy předmět BI-PGA. Předtím úspěšně absolvoval předmět BI-MGA, bohužel opakuje lineární algebru, o které ví, že se používá i v počítačové grafice. Proto se chce více připravovat během semestru.

Typické použití programu: Učitel po svém výkladu ohledně transformací předá studentům scénu na procvičení. Ve scéně jsou náhodně rozmístěny tři kostky, řekne jim aby je bez přidání matice dali nad sebe. Ten kdo bude mít hotovo se má přihlásit a učitel pak bude chtít vysvětlit, jak, co a kde změnil.

4.2.4 Programátoři

Tato skupina uživatelů vytváří v programu *I3T* složitější scény. Tyto scény mohou vznikat jako tutoriál pro ostatní, nebo simulovat nějakou situaci, se kterou si v jiném projektu nevědí rady. Potřebují mít pracovní plochu co nejpřehlednější a chtějí, aby se jim vešlo co nejvíce ovládacích prvků do okna a nemuseli ho tak zvětšovat, případně popojíždět po ploše. Chtějí mít náhled scény na jednom monitoru a pracovní plochu na druhém.

Persona: Bc. Petr Novotný Petr je 24letý student FEL ČVUT v Praze. Prvním rokem studuje magisterský obor počítačová grafika, z kterého už má bakalářský titul. Petr rád vytváří 3D hry. Chvíli pracoval i v herním průmyslu, má tedy velké zkušenosti s transformacemi. Program *I3T* vyzkoušel a občas ho používá pro simulaci nějakých složitějších věcí, které by měl programovat nebo když potřebuje něco ukázat kamarádům.

Typické použití programu: Je potřeba vytvořit scénu pro vysvětlení gimbal locku pro použití do výukového videa. Petr si to dal za úkol a začal pracovat. Má otevřené okno s pracovní plochou na jednom monitoru a náhled scény na druhém. Ke všem maticím přikládá komentáře, proč a jak je vytvořil. Poté ještě vytvoří tutoriál, aby uživatel věděl, jak má změnit hodnoty, aby se dostal do gimbal locku a ztratil tak jeden stupeň volnosti.

4.2.5 Kdokoli - náhodný uživatel se zájmem o grafiku

Tato skupina je nejvíce obecná. Uživatelé mohou mít různé cíle, někteří program budou chtít použít, aby se naučili/zopakovali lineární transformace, nebo naopak to mohou být nadšenci do počítačové grafiky a mohou do programu tvořit jednotlivé scény a tutoriály a vylepšovat tak celý program. Určitě budou chtít, aby používání programu bylo co nejintuitivnější. Budou chtít lehce najít řešení jejich problému například v nápovědě.

Budou zkoušet jednotlivé tutoriály a hodnotit jejich smysluplnost a přínosnost při učení. Mohou si vytvářet scény pro simulaci jejich problému. Pokud to budou programátoři, mohou si v programu nasimulovat jejich problém a poté si vygenerovat jednotlivé matice jako C++ kód.

Ti kteří budou tvořit jednotlivé tutoriály, chtějí jednoduchý nástroj pro jejich tvorbu a mít scénu co nejpřehlednější.

Persona: Josh Parner je 45letý americký vědec, který se zabývá vývojem nových technologií pro virtuální realitu. Vystudoval MIT. Čte několik odborných webů zabývajících se počítačovou grafikou. Rád tvoří učební materiály pro ostatní. Je členem několika internetových fór, kde velmi často pomáhá lidem řešit jejich problémy ohledně počítačové grafiky.

Alternativní persona: Roman Malý je 25letý programátor v herním studiu. Vystudoval bakaláře na FIT VUT v Brně. Velmi rád hraje hry a zajímá se o nové herní systémy, proto sleduje několik webů ohledně počítačové grafiky. Rád chodí s přáteli na různé kulturní akce. Roman nerad čte knihy. Při učení nových věcí si radši věci vyzkouší v praxi, namísto studia teorie.

Typické použití programu: Na webu čte článek, který se zabývá problémem gimbal lock. V článku je odkaz na program *I3T*, ve kterém si díky tutoriálu může problém vyzkoušet. Proto si program stáhne a projde postupně tutoriálem zabývajícím se problémem gimbal lock.

4.3 Testování klávesových zkratek

V předešlém testování provedeném v rámci předmětu A4B39TUR v roce 2017 [30] (zpráva z testování je na CD), bylo zjištěno, že program používá nestandardní klávesové zkratky. Proto jsem vytvořil tabulku viz tabulka 4.1 jednotlivých funkcí programu a často používaných zkratek pro funkce v podobných programech, z nichž sestavím dotazník, z něhož zjistím, která klávesová zkratka vyhovuje uživatelům nejvíce a jestli by využili vlastní nastavení zkratek.

V úvodu dotazníku zmíním, že jde o 3D program a poprosím uživatele pracující s 3D programy, aby hlasovali pro zkratky, které očekávají v 3D programu. V úvodu je zmíněno, že pokud respondent danou zkratku nepoužívá, nemusí na tuto otázku odpovídat.

Na začátku dotazníku jsem se respondentů zeptal s jakými 3D programy mají zkušenosti a jak často používají klávesové zkratky, což mi pomůže ur-

Tabulka 4.1: Tabulka funkcí a jejich klávesových zkratek

Název zkratky	1. Varianta	2. Varianta	3. Varianta	4. varianta
Kopírování do schránky	Ctrl + C	Ctrl + táhnutí myší	C	Ctrl + Ins
Vkládání ze schránky	Ctrl + V	V	Shift + Ins	-
Vyjmutí do schránky	Ctrl + X	X	-	-
Duplikace	Ctrl + D	Ctrl + J	Ctrl + kliknutí levým tlačítkem myši	-
Smazání	Del	X	Backspace	Ctrl + U
Krok zpět v historii	Ctrl + Z	B	Ctrl + B	Alt + Backspace
Krok vpřed v historii	Ctrl + Shift + Z	Ctrl + Y	N	-
Výběr objektu	klik levým tlačítkem na objekt	dvojitý klik levým tlačítkem na objekt	klik pravým tlačítkem myši	-
Výběr více objektů	Ctrl + stisk levého tlačítka myši	stisk levého tlačítka myši a táhnutí myší	stisk pravého tlačítka myši a táhnutí	Ctrl + stisk levého tlačítka myši a táhnutí myší
Vybrat vše	Ctrl + A	dvojité kliknutí levým tlačítkem	trojité kliknutí levým tlačítkem	A
Zrušení výběru	Esc	A	kliknutí levým tlačítkem mimo výběr	Ctrl + D
Přiblížení	otáčení kolečka myši	Ctrl + otáčení kolečka myši	Alt + otáčení kolečka myši	-
Rotace kamery	stisk kolečka myši + pohyb myší	Alt + stisk kolečka + pohyb myší	stisk pravého tlačítka myši + pohyb myší	stisk levého tlačítka myši + pohyb myší
Pohyb kamery	Shift + stisk kolečka myši + pohyb myší	stisk kolečka myši + pohyb myší	-	-
Uložit	Ctrl + S	S	-	-
Uložit jako	Ctrl + Shift + S	Shift + S	-	-

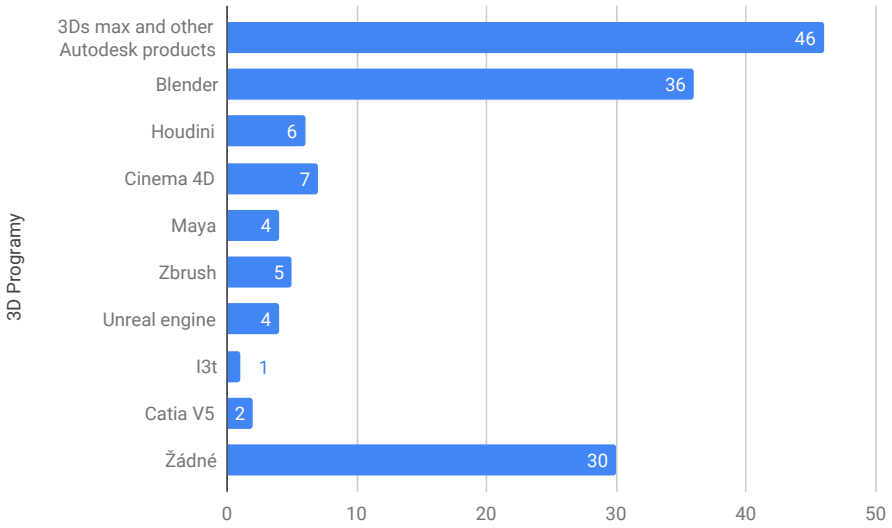
čit důležitost jejich správnosti a případně zvolit zkratky podle nějakého programu, jenž respondenti často používají.

Do dotazníku jsem se rozhodl nepřidávat možnosti zkratek pro změnu pohledu, přidalo by to moc otázek a v *I3T* tato funkce není často používána. Jelikož je program vyvíjen pro operační systém *Windows* jsou uvedeny v dotazníku klávesové zkratky pro tento systém.

4.3.1 Výsledky dotazníku na klávesové zkratky

Českou verzi dotazníku vyplnilo 77 respondentů, přičemž 36 odpovědí bylo od lidí z fakulty informačních technologií (odkaz jsem sdílel jen na fakultních facebookových skupinách). Jelikož nebylo povinné odpovědět na všechny otázky, pohybuje se počet odpovědí u jednotlivých otázek mezí 50-70 odpověďmi. Podobu dotazníku si můžete prohlédnout na odkazu: `https://forms.gle/vvdR7DrhTTwyti8F8`.

Za pomoci prof. Bedřicha Beneše byl dotazník vyplněn i na universitě v Purdue (*Purdue university*). Anglické verze se zúčastnilo 39 studentů, tedy celkový počet respondentů je 116. Výsledné grafy budou tvořeny ze spojení těchto dvou dotazníků. Soubory s daty s jednotlivých dotazníků a ze spojení obou dotazníku jsou přidány na CD.



Obrázek 4.1: Oblíbené 3D programy testovaných uživatelů

4.3.1.1 Jaké 3D programy často používáte a znáte jejich klávesové zkratky?

U první otázky jsem zjišťoval jaké 3D programy respondenti používají, tato otázka mi poté může pomoci k další analýze dalších odpovědí. Na fakultě informačních technologií je v několika předmětech používán program *Blender*, proto není překvapením, že zde má velké zastoupení a to 42 % (počítáno z celkového počtu respondentů – 117) viz graf 4.1. Američtí studenti mají v oblibě produkty od společnosti *Autodesk* a to hlavně *3Ds Max* a *Maya*, který v americkém dotazníku měl 33 odpovědí tedy 86,8 %. Celkově mají produkty od firmy *Autodesk* zastoupení 46 odpovědí tedy 53,8 %.

V druhé otázce jsem se uživatelů ptal, jak často používají klávesové zkratky. Otázka byla ve škále 0–5, kde pět znamenalo stále nebo téměř stále. 52 % uživatelů zvolilo možnosti 4 a 5. Pro možnost tři hlasovalo 21 % respondentů. Z toho můžeme usoudit, že klávesové zkratky jsou pro uživatele velmi důležité a jejich správné nastavení velmi usnadní ovládání programu.

4.3.1.2 Funkce myši a klávesové zkratky ve 2D

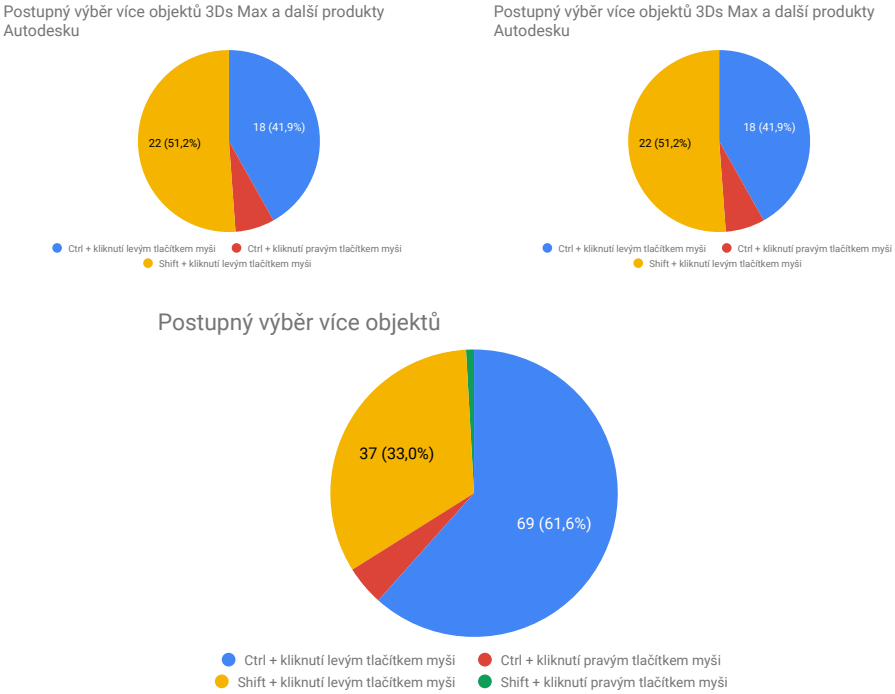
Zkratky pro kopírování (Ctrl + C), vložení (Ctrl + V) a vyjmutí do schránky (Ctrl + X) mají procentuální zastoupení odpovědí více jak 90 %. U zkratky smazání má zkratka (Del) 84 %, druhá v pořadí je klávesa Backspace s 11 %. U mazání stojí za úvahu, jestli neimplementovat obě zkratky. Uživatelům by to nemělo vadit a pokrylo by se tak 95 % respondentů dotazníku.

Větší rozdíly nastaly až u zkratek pro duplikaci, kde největší hodnocení 59 % má zkratka Ctrl + D druhá zkratka – Ctrl + kliknutí levým tlačítkem myši má 23 %. U těchto zkratek by mělo jít vyhovět oběma skupinám uživatelů a do programu vložit obě možnosti a pokrýt tak 82 % uživatelů.

Zkratka pro krok zpět v historii (*redo*) má 90 % odpovědí pro Ctrl + Z. U zkratky pro krok vpřed v historii odpovědi tak jednoznačné nejsou, zkratka Ctrl + Y 47,5 % a zkratka Ctrl + Shift + Z 47,5 %. Každá zkratka měla po 48 odpovědích a dohromady tyto zkratky pokryjí 95 % respondentů.

Snažil jsem se vyhledat, kde vznikl problém u těchto dvou zkratek. V jedné z mnoha diskuzí na internetu [31] říkají, že Ctrl + Y je používána často v programech vyvíjených hlavně pro *Windows* nebo přímo touto společností. Zkratka Ctrl + Shift + Z, respektive Comand + Shift + Z vznikla v operačním systému *Mac OS* a poté jí zahrnula firma *Adobe* do svých programů a dostala se tak i mezi uživatele operačního systému *Windows*. Mnoho aplikací podporuje obě zkratky pro krok zpět, aby tak vyšly vstříc co největší skupině uživatelů.

Při zkoumání korelace u zkratky zpět v historii v závislosti na oblíbeném programu u *Blenderu* i *3Ds Max* v mém dotazníku vyšly podobné výsledky, kde měly obě dvě zkratky kolem 50 % odpovědí, tedy jsem nenalezl žádná kauzalita.



Obrázek 4.2: Dolní graf zobrazuje všechny odpovědi respondentů. Ve zbývajících můžete vidět odpovědi v závislosti na oblíbeném programu (*Blender/3Ds Max*)

U výběru objektu zvolilo 92 ze 112 respondentů (82 %) možnost kliknutí levým tlačítkem na objekt, kterou jsem zvolil pro program *I3T*. Po devíti procentech mají možnosti kliknutí pravým tlačítkem (tato zkratka se vyskytovala u respondentů, kteří měli zkušenosti s programem *Blender*) a dvojité kliknutí levým tlačítkem na objekt.

U výběru více objektů má kliknutí levým tlačítkem a tažení myší 80,9 % odpovědí. Je to tedy jednoznačná volba. Další odpověď s devíti procenty je kliknutí pravým tlačítkem a tažení myší. Za zmínění stojí ještě zkratka Ctrl + kliknutí levým tlačítkem myši + tažení myší, která má 8,2 %.

U postupného výběru více objektů měla zkratka Ctrl + kliknutí levým tlačítkem myši 61,6 % a Shift + kliknutí levým tlačítkem myši 33 %. Dvacet z třiceti sedmi odpovědí pro zkratku s klávesou Shift bylo od amerických respondentů, u kterých vím, že používají více produkty od firmy *Autodesk*.

Proto jsem začal zkoumat korelace této zkratky z oblíbeným programem viz Obr. 4.2 a byly nalezeny korelace pro zkratku s klávesou Shift u *3Ds Max* a s klávesou Ctrl u programu *Blender*. Jak je vidět z grafu jednotlivé zkratky korelují s programy. V obou programech se používají pro výběr více objektů obě klávesové zkratky a to bez rozdílu. V Blenderu se dokonce varianta

s klávesou Ctrl používá až při výběru v editačním módu, proto jsem kauzalitu zamítl.

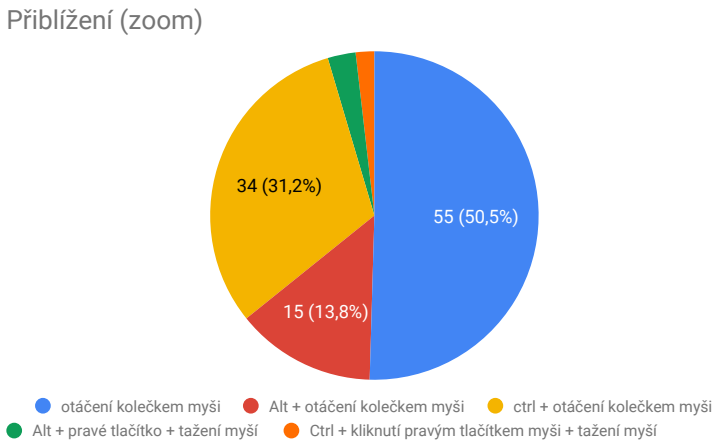
Doporučuji implementaci obou zkratek pro pokrytí 94,6 % uživatelů. Většina uživatelů zná rozdíl mezi těmito klávesami z výběru sloupce, kdy se s klávesou Shift vybírá celé rozmezí a s Ctrl jen ty soubory, na které uživatel klikne. Jednotlivé klávesy by se mohly případně odlišit při nutnosti nějakého speciálního výběru, ale využití této funkce v *I3T* mě nenapadá.

U funkce vybrat vše, má nejvíce odpovědí zkratka Ctrl + A s 81,9 %. Další odpověď z 11,4 % je dvojité kliknutí levým tlačítkem myši. Do programu navrhuji na základě dotazníku implementovat jen první zmíněnou zkratku.

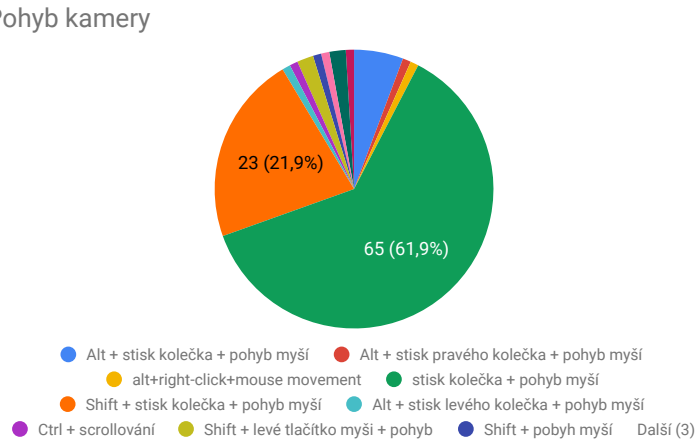
U zrušení výběru kde možnost Esc má 41,8 % a kliknutí levým tlačítkem myši 47,3 %. Zde doporučuji v programu použít obě varianty zkratek pro pokrytí co nejvíce uživatelů. Implementace obou zkratek by neměla být náročná a ovladatelnost programu se tím značne zlepší. Správnost použití více zkratek pro jednu funkci mi potvrdil doc. Ing. Zdeněk Míkovec, Ph.D.

4.3.1.3 Funkce pro práci ve 3D náhledu

Problém nastává u zkratek pro pohyb v 3D prostoru, u těchto zkratek je potřeba zvolit jen jednu variantu zkratky. V 2D programech pro příklad v prohlížečích je mezi uživateli známá zkratka přiblížení (*zoom*) – Ctrl + otáčení kolečka myši, protože samotné točení kolečka se používá pro posouvání po stránce. Ve 3D programech se nepotřebujeme posouvat, a proto většina programů používá otáčení kolečka a klávesou Shift , Ctrl nebo Alt se mění jen rychlost přibližování.



Obrázek 4.3: Graf – přiblížení (*zoom*), na tuto otázku odpovědělo 109 respondentů



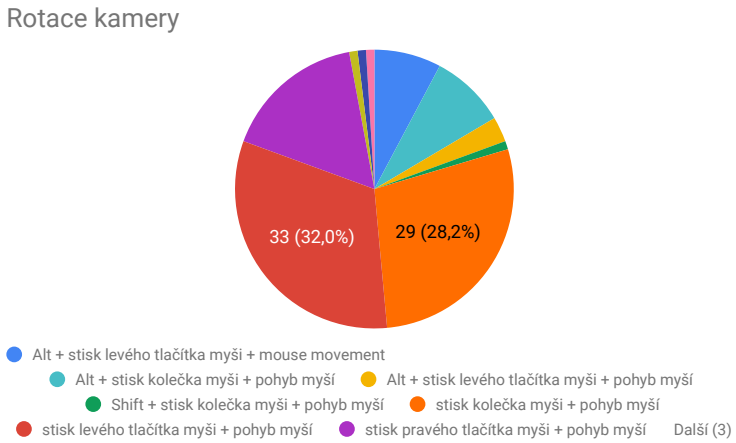
Obrázek 4.4: Graf – pohyb kamery

Přestože v dotazníku bylo uvedeno, aby uživatelé uvedli zkratku pro danou funkci, kterou používají v 3D programech, tak se v dotazníku viz graf 4.3 objevuje dost často varianta známá z prohlížečů (Ctrl + otáčení kolečkem myši), která má 31 %. Přibližně stejná procenta má i při vybrání uživatelů programů *Blender* a *3Ds Max*. Varianta používaná v 3D prostředí (otáčení kolečkem) má 50,5 %. Na základě výše zmíněných informací navrhuji použít pro přiblížení zkratku otáčení kolečka.

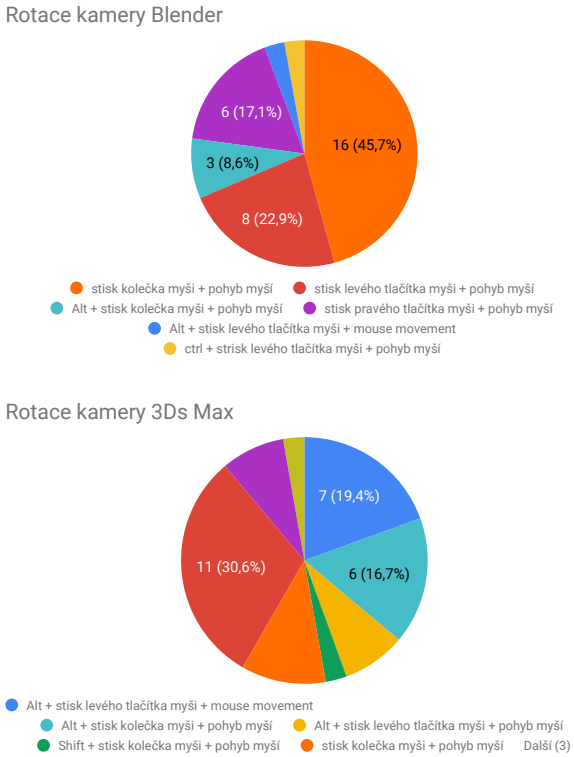
Následující dvě zkratky pro funkci rotace kamery viz graf 4.5 a pohyb kamery viz graf 4.4 musím rozebrat dohromady, protože z dotazníku vyplynulo, že funkce mají stejnou klávesovou zkratku – stisk kolečka + pohyb myší, která má u rotace kamery 28,2 % (druhé místo hned za stisk levého tlačítka + pohyb myší s 32 %) a u pohybu kamery 61,9 %, kde na druhém místě je zkratka Shift + stisk kolečka + pohyb myší. Za těchto okolností navrhuji, pro pohyb kamery použít zkratku stisk kolečka + pohyb myší.

U zkratky pro rotaci kamery je to složitější. Na třetím místě s 16,5 % je stisk pravého tlačítka + pohyb myší. Zkoušel jsem najít korelaci mezi oblíbenými programy a u *Blenderu* viz graf 4.6 má zkratka stisk kolečka + pohyb myší 45,7 %, tedy koreluje a našel jsem tak kauzalitu s programem, protože se v programu *Blender* k rotaci tato zkratka používá. Naopak u *3Ds Max* žádná korelace u zkratky pro rotaci není, jak je vidět v grafu 4.6.

Tedy nám zbývají zkratky s nejvíce odpověďmi a to stisk levého tlačítka + tažení myší tato zkratka se v 3D editorech používá k výběru objektu, proto v nich nemůže být použita pro rotaci. Protože se v *I3T* ve 3D scéně žádné výběry nedělají, navrhuji použít zkratku stisknutí levého tlačítka + tažení myší. Pokud by měla být do programu přidána funkce pro výběr, navrhuji zkratku umístit na pravé tlačítko namísto levého.



Obrázek 4.5: Graf – rotace kamery



Obrázek 4.6: Graf uživatelů programu *Blender* (nahoře) a *3Ds Max* (dole) k otázce rotace kamery

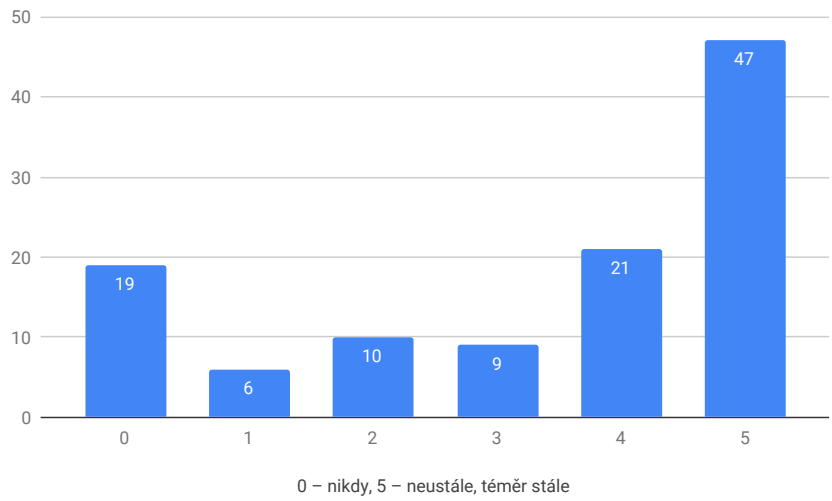
4.3.1.4 Funkce pro ukládání souborů a jejich klávesové zkratky

V dotazníku jsem se ptal na zkratku pro funkci uložit, u které má nejvíce odpovědí zkratka Ctrl + S s 95,4 % a u funkce uložit jako má nejvíce hlasů zkratka Shift + Ctrl + S 85,3 %. Za zmínění stojí i druhá zkratka v pořadí, kterou zvolilo 10,8 % respondentů. U obou funkcí navrhuji použít výše zmíněné zkratky s nejvíce hlasy.

V aktuální verzi programu nejsou výše zmíněné zkratky implementovány, zařadil jsem tedy do dotazníku otázku – „Jak často uživatelé používají zkratku pro ukládání?“ Čtyřicet sedm ze sto třinácti respondentů odpovědělo, že zkratky pro ukládání používají velmi často. Jak je vidět v grafu 4.7 uživatelé zkratky pro ukládání používají poměrně často. Navrhuji tedy jejich přidání do programu. Implementace této funkcionality by neměla být náročná.

4.3.1.5 Výsledky dotazníku

Z mého dotazníku vyplynulo, že testovaní uživatelé nemají jedno-klávesové zkratky v oblibě. Před rychlostí a efektivitou má pro většinu uživatelů větší význam zvyk. Jedno-klávesové zkratky jsou ochotní se naučit pouze u často používaného programu. V dotazníku byla většina zvolených zkratek více-klávesová. Výjimku tvoří zkratky pro pohyb ve 3D, kde to jsou sice jedno-klávesové/tlačítkové zkratky, ale všechny tyto zkratky jsou tlačítka myši.



Obrázek 4.7: Graf používání klávesové zkratky pro ukládání

V některých programech jsou jedno klávesové zkratky hojně používány. Jedno klávesovými zkratkami a způsobem zmáčknutí jednotlivých kláves se zabývá velmi zajímavý výzkum Finger Aware Shortcuts [32]. Autoři zde zmiňují, že uživatelé jsou zvyklí na určité klávesové zkratky a pokud lze pro jednu funkci implementovat více klávesových zkratek, tak to velmi doporučují. Jejich výzkum vznikl pro urychlení práce v různých programech, což by mělo být používání jedno-klávesových zkratek. V některých programech jsou tyto zkratky velmi využívány, v těchto programech se pomocí nich přepínají různé módy a jsou zde i různé stavy programu, ale i tyto programy používají více-klávesové zkratky. Autoři zmiňují programy jako *Adobe Photoshop*, kde se pomocí výše zmíněných zkratek přepíná například mezi výběrem, štětcem nebo gumou. V programu *Blender* jsou jedno-klávesové zkratky použity pro funkce jako je zvětšení S (*scale*), posun G, pro změnu pohledů atd.

4.4 Testování uživatelského rozhraní

4.4.1 Pluralistické průchod

Původně jsem zamýšlel otestovat program za pomoci uživatelského testování s aktuální verzí programu a mým návrhem tutoriálu, který bych dal testovaným uživatelům k dispozici na papíře a řídili by se instrukcemi v tutoriálu. Při konzultaci s docentem Schmidtem jsem se dozvěděl, že tato varianta by byla pro uživatele velmi matoucí a poradil mi, ať udělám test pomocí pluralistického průchodu s papírovou verzí tutoriálu z mého návrhu o deseti stránkách.

Doporučil mi udělat pilotní test jen se třemi uživateli, moderátorem a vývojovým týmem *I3T* v dostatečně velké učebně. Při každé stránce si uživatelé napíší svůj postup na papír nebo do připraveného dotazníku, jenž by po určitém počtu úkolů (viz. příloha scénář pluralistického testování) konzultovali s vývojovým týmem a ten by okamžitě přišel na nedostatky programu. Ostrý test by měl být z 5–9 uživateli.

4.4.1.1 Výsledek pluralistického testování

Pluralistické testování proběhlo 1. 4. 2019 v učebně FIT ČVUT. Testování se zúčastnil vývojový tým a čtyři respondenti, další dva se omluvili. Samotné testování trvalo hodinu a většinu tohoto času zabrala diskuze.

První problém, na který uživatelé upozornili hned po prvních čtyřech stránkách, byla nejasnost názvů výstupů a vstupů sekvence a celkové nepochopení její funkce. Testovaní vůbec nevěděli, že matice se mezi sebou v sekvenci násobí. V diskuzi nakonec vývojáři a respondenti dospěli k tomu, že tento problém by se vyřešil přidáním křížku mezi jednotlivé matice v sekvenci.

Názvy vstupů a výstupů byly pro uživatele velmi matoucí. Testovaní nebyli schopní si pod nimi představit co jednotlivé vstupy a výstupy znamenají. Pro vstup (parent) a výstup (child) by raději respondenti měli samotný znak křížku

(místo bílého kolečka) případně doplněný o název – matice (*matrix*). Výstup, který se v návrhu jmenoval matice (*matrix*) by nazvali show, protože jim bylo vysvětleno, že v něm je uložena výsledná matice, která vznikne vynásobením všech matic v napojených sekvencích. Pro výstup z názvem (*storage*) uživatelé po diskuzi navrhli název (*copy*), protože z tohoto výstupu by měla být předána matice, jenž je výsledkem násobení matic uvnitř vybrané sekvence. Změny názvů budou přidány do prototypu a znovu otestovány.

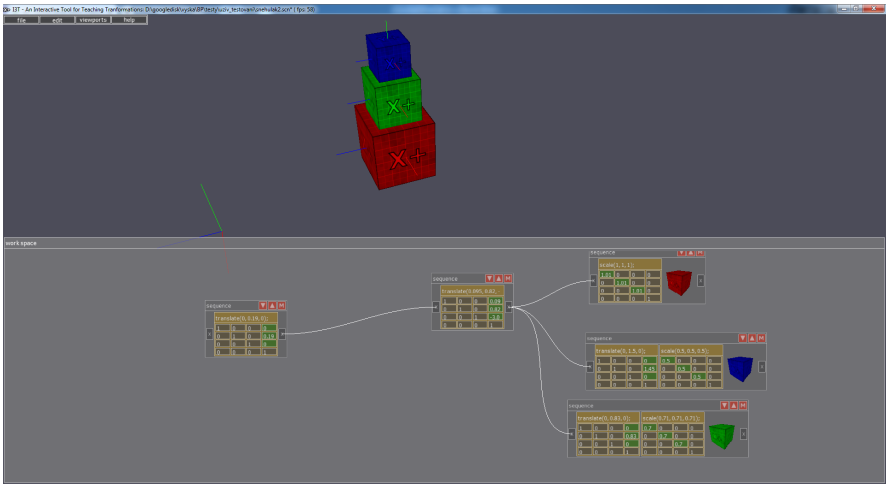
Bylo zjištěno, že jednotlivé sekvence by měly jít přejmenovat. Bylo zmíněno, že pod názvem „sekvence“ si uživatel velmi těžko představí k čemu slouží. Byl navržen název blok matic (*block of matrices*). Na grafice krabičky by respondenti změnili tlačítko pro zmenšení z trojúhelníkové tvaru na klasické obdélníkové tlačítko. Testovaní navrhli i přidání křížků pro odstranění sekvence. Vývojáři si tímto krokem nebyli jistí, protože poté hrozí překliknutí a smazání sekvence.

V diskuzi se vývojáři ptali, jestli by respondenti oddělili objekt od bloku matic (dříve sekvence). Všichni respondenti s tímto krokem souhlasili a pro objekt tak vznikla samostatná krabička, ke které se připojí blok matic.

Součástí formuláře, který respondenti vyplňovali byla otázka, jestli chtějí mít při otevření programu prázdnou scénu nebo vložený základní objekt. Z diskuze vyplynulo, že preferují prázdnou scénu. Scénu s objektem by ocenili jen v případě, že s programem začínají. Tuto scénu chtěli také proto, že se na ní dají rychle otestovat základní transformace. Vývojáři jim vysvětlili, že program bude obsahovat připravené scény na různá témata. Nakonec se všichni shodli na prázdné scéně.

Dále bylo navrženo, že tutoriály by neměly být tříděny nejen podle hodnocení, ale i podle obtížnosti (začátečník, pokročilý, expert). Jinak se uživatelům návrh úvodní strany s výběrem tutoriálu líbil. Jediné co bylo vytknuto tutoriálu, byly dlouhé texty. Ty si ale vývojový tým obhájil tím, že je to výukový program a musí zde být vysvětlena látka současně s postupem. Po diskuzi dospěli účastníci testování k tomu, že s obtížností tutoriálu, bude textu v tutoriálu ubývat.

Po poradě s vedoucím práce jsme se rozhodli další pluralistický test neprovádět a soustředit se na přípravu prototypu pro uživatelské testování. Dále jsme zjistili, že diskuze jen se třemi účastněnými byla i tak velmi dlouhá. Nedokázali jsme si představit jak by test probíhal s pěti až osmi uživateli a jestli by se dokázali uživatelé při diskuzi shodnout a nevznikalo by mnoho nápadů, mezi kterými bych musel vybírat. Problémem byla synchronizace účastníků, aby pracovali na stejném čísle obrázku, pro příště bych doporučil někam umístit aktuální číslo strany, na které se aktuálně všichni účastníci nachází a o které zrovna probíhá diskuze.



Obrázek 4.8: Scéna v aktuální verzi *I3T* na jejímž základě byl vytvořen tutoriál v prototypu

4.5 Příprava prototypu k uživatelskému testování

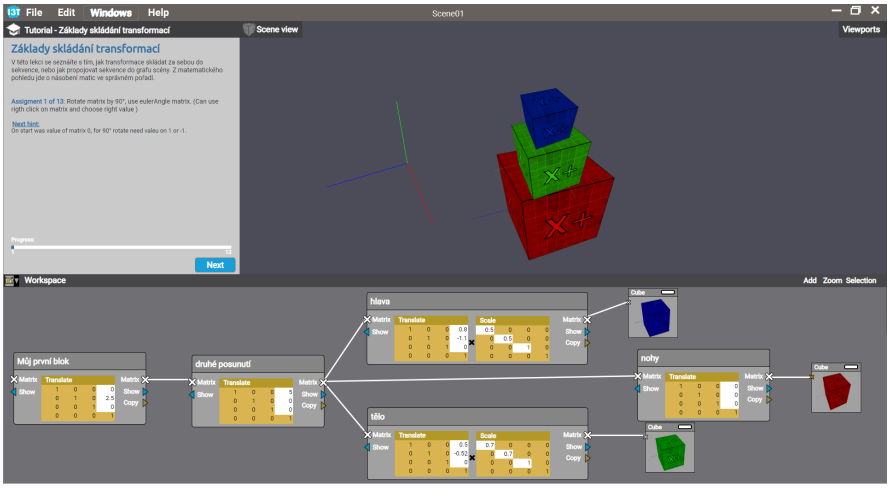
Jelikož implementace navrhovaných změn do aktuálního programu je velmi časově náročná, rozhodl jsem se vytvořit prototyp vycházející z mých návrhů, na kterém bych otestoval nové funkce a vzhled programu.

Na výrobu prototypu jsem použil program *Axure* [33], který poskytuje studentskou licenci zdarma. Tato aplikace umožňuje tvorbu prototypů formou webových stránek. V programu sie postupně sestaví návrh a implementuje potřebná logika a poté se vygeneruje kód jako HTML stránka nebo prototyp rovnou zveřejní na webu *Axure Share*[3] (k tomuto kroku stačí být přihlášen v programu pod svým účtem axure a kliknout na tlačítko sdílet(*share*)).

S programem jsem před tvorbou prototypu neměl žádné zkušenosti. Naštěstí *Axure* má velmi dobře zpracované stránky s řešením jednotlivých problémů i s přidanými demo ukázkami. Jelikož tento program využívá mnoho firem je kolem něho rozsáhlá komunita, která přispívá hlavně na *Axure fórum*[4], kde lze najít řešení mnoho problémů i s přiloženými projekty řešícími daný problém.

Po zveřejnění projektu na webu je k samotnému prototypu připojen boční panel, který obsahuje stromovou strukturu webové stránky, poznámkový blok, prostor pro diskuzi a výstup z konzole, kde jsou vidět jednotlivé akce uživatele a hodnoty proměnných. Lze zde i nastavit jednotlivé proměnné na počáteční hodnotu. [34] Bohužel jsem nepřišel na to, jak výstup z konzole uložit, takže je přístupný jen po dobu testování, dokud uživatel neobnoví stránku (lze po testování zkopírovat výstup z konzole do souboru, ale bude to jen čistý text bez

[3] `https://share.axure.com/`
[4] `https://forum.axure.com/`



Obrázek 4.9: Výsledná scéna tutoriálu sněhuláka v prototypu *I3T*

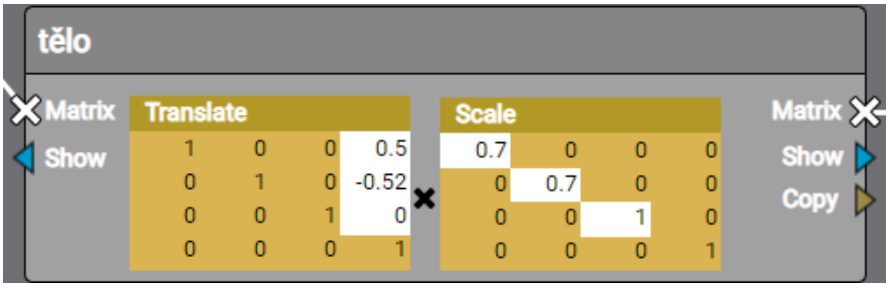
formátovaní). Konzole je velmi užitečná při vývoji prototypu, protože vývojáři ukáže, co vše se vyvolalo při dané akci.

Už od začátku vývoje prototypu jsem věděl, že v programu *Axure* nebudu moci vytvořit 3D náhled. Z tohoto důvodu jsem se rozhodl sestrojit zjednodušený prototyp, aby na něm bylo možné udělat tutoriál, který připravili páni Felkel a Pilka pro starou verzi programu *I3T* [35]. Vybral jsem si část, kde se ze tří kostek postaví „sněhulák“. Tento tutoriál byl poupraven. Uživatel tvoří celou scénu nově a liší se hodnoty jednotlivých matic. Bohužel vše je jen simulace 3D transformací, proto hodnoty matic ve výsledné scéně viz Obr. 4.9 neodpovídají skutečným hodnotám matic, které jsou vidět na obrázku z aktuální verze programu *I3T*, kde je vypracována scéna podle tutoriálu ke starší verzi programu viz Obr. 4.8. Text jednotlivých stran tutoriálu naleznete v příloze.

Ve skutečnosti je náhled scány jen 2D a bod (0,0) je v levém horním rohu okna s náhledem scény (*scene view*). Jednotlivé hodnoty v maticích se sčítají a poté objekt posouvají nebo mění velikosti v osách x a y. Osa z je úplně ignorována, protože jediná možnost kterou by mohla plnit, je měnit z-index (výšku prvku na ose z), ale to by uživatele ještě více zmátlo.

Implementovat celou logiku programu v prototypu by bylo velmi náročné a vlastně i nepotřebné. *Axure* je program, který slouží k rychlému návrhu a programování v něm je velmi složité. Vytvořit složitější logiku je náročné a celý návrh se tím stává nepřehledným. Logika je řešena za pomoci podmínek (if a elif). Jejichž rozhodovací funkce mohou obsahovat buď logickou operaci *or* nebo *and*, ale nelze je mezi sebou v rámci jedné podmínky kombinovat, což velmi komplikuje vytváření pokročilejších funkcí.

V *Axure* se jednotlivé akce (např. schovat, označ, přemísti, zobraz, atd.) vyvolávají na základě podmínek a jejich logiky, kterou jsem poznal výše. Podmínky jsou vždy přiřazeny k nějaké události (*event*) (např. OnClick, OnMou-

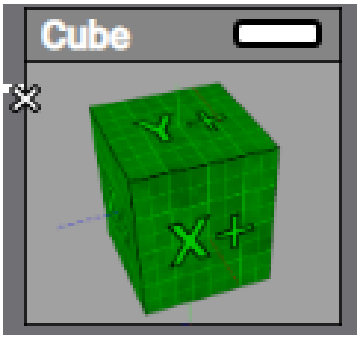


Obrázek 4.11: Vzhled bloku matic v prototypu po pluralistickém testování

seEnter, OnDrag, atd.), které jsou shodné jako událostmi v programovacím jazyce javascript. Jedna událost může mít více podmínek a pořadí jednotlivých akcí lze změnit.

Do prototypu jsem přidal následující změny zjištěné v pluralistickém testování. Byly přejmenovány sekvence na bloky matic (block of matrices) a vstupy a výstupy sekvencí (*storage* na *Copy*, *matrix* na *Show* a vstup a výstup pro násobení jsem z *parent/child* změnil na *matrix*). Objekt byl oddělen od bloku matic (dříve sekvence) viz Obr. 4.10. Mezi matice v bloku matic byl přidán křížek, který zdůrazňuje, že se jednotlivé matice násobí viz Obr. 4.11. Všechny krabičky přidané do pracovní plochy (*workspace*) se dají přejmenovat. Při implementaci jsem narazil na problém pro generování čar spojující jednotlivé krabičky, pokud se s nimi pohybovalo čáry zůstaly na původních místech, proto v prototypu v pracovní ploše je pozice jednotlivých krabiček a čar pevně daná.

V prototypu lze zobrazit všechny položky jednotlivých menu, ale jen některé položky jsou implementovány (kliknutí na ně vyvolá nějakou akci), implementace všech funkcí by byla velmi složitá a některé by nešly s danou technologií vytvořit.



Obrázek 4.10: Vzhled krabičky pro objekt

4.6 Testování použitelnosti

4.6.1 Kognitivní průchod

Prototyp jsem chtěl mít v co nejlepším stavu, než začne uživatelské testování. Oslovil jsem proto několik osob a provedl kognitivní průchod. Cílem průchodu bylo splnit tutoriál o základních transformací, připravený v prototypu. Po splnění tutoriálu dostali testeři za úkol smazat spoj a blok matic.

Uživatelé měli jednotlivé pokyny přímo v programu, takže bylo potřeba jenom respondenty seznámit s prototypem a zadat jim úkol, tedy splnění tutoriálu na základní transformace. Respondenti byli upozorněni na hůře fungující rozbalovací menu, která bohužel v programu nejdou lépe vytvořit. Dále bylo řečeno, že se jedná o prototyp, který simuluje 3D prostředí, ale pracuje jen s 2D obrázky a se souřadnicovým systémem, který má střed v levém horním rohu.

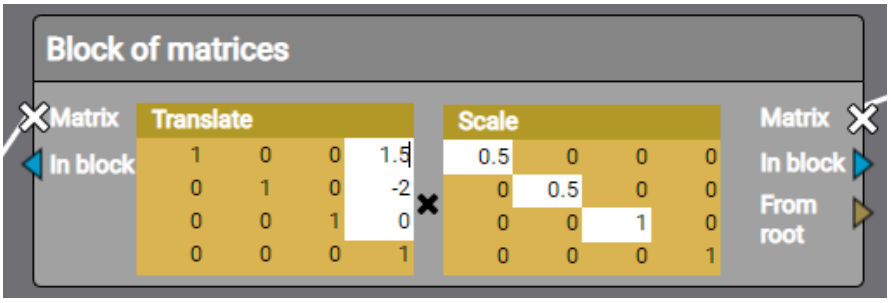
Testování se zúčastnili čtyři respondenti, každý tak zastoupil jednu skupinu uživatelů mimo specifické skupiny studentů na cvičení. Skupinu studentů pro samostudium a skupinu studentů na cvičení jsem pro toto testování sloučil do jedné. Tři skupiny byly zastoupeny respondenty, kteří do této skupiny patří, takže nebylo nutné použít personu. Jediná skupina u které musel expert použít personu, byla skupina student.

U všech testerů nastal problém hned u druhého úkolu, kde si vůbec nevšimli modrou barvou zvýrazněných položek v menu pro matici posunutí, které jim mělo pomoct tuto matici nalézt. Matici hledali v menu operátoru a modrá barva, která zdůrazňovala položku transformace (*transformations*) si nevšimli. Tato chyba byla opravena, takže nyní se rozsvítí modře i menu přidat (*Add*).

Další chyba (jen v rámci prototypu) vyskytující se u všech testovaných byla vytváření spoje pomocí přetažení . Na konci testu bylo všem vysvětleno, že v plné verzi půjde takto spoj vytvořit, ale do prototypu tato funkce nebyla přidána z důvodu její náročnosti na implementaci.

Expert zastupující skupinu učitelů poukázal na špatné názvy vstupů a výstupů *show* a *copy* u bloku matic. Po diskuzi s vývojovým týmem a ostatními experty byli názvy vstupů bloku matic změněny v prototypu – *show* na *in block* a *copy* na *from root* viz Obr. 4.12.

Experti zastupující skupiny učitelů a programátorů poukázali na matoucí obrázek středu světového souřadnicového systému uprostřed prototypu. Na začátku fungoval jako střed (byly jsem vloženy kostky), ale poté uživatele mátl a nedokázali pořádně měnit hodnoty v maticích, protože si mysleli, že střed je v tomto obrázku. Vlastně uživatelé si mysleli, že pracují s 3D prostředím i náhledem, ale ve skutečnosti to byl 2D náhled (během dalších testování na



Obrázek 4.12: Podoba bloku matic po změně názvů

to byli upozorněni). Tento obrázek byl nahrazen osami x a y, které jsou vlevo a nahoře. Pomohou tak respondentům zjistit kde se nachází počáteční bod.

Testeři zastupující náhodného uživatele a programátora měli problémy se spojením dvou bloků matic. Nevěděli, který vstupů a výstupů měli propojit. Tento problém se vyřešil změnou textu tutoriálu. Nyní je u prvního spojování dvou bloků matic zřetelně popsáno, jaké vstupy a výstupy se mají spojit. Tentýž problém nastal u připojování bloku matice k objektu, kde byl zvětšen symbol křížku, aby byl více viditelný.

Dále testeři našli některé chyby v textu, jako bylo používání starého názvu sekvence místo blok matic. Tyto nepřesnosti byly opraveny.

Tester představující skupinu náhodného uživatele ocenil, že fungovalo přeskakování mezi otevřenými buňkami pomocí klávesy Tab, což uživatelům může ušetřit mnoho času.

Se smazáním spoje nebo bloku matic neměli testovaní žádný problém. Jen si stěžovali, že po přerušení spoje se kostka nepřesune do počátečního bodu tedy vlevo nahoru. O tomto problému jsem věděl a rozhodl jsem se ho pro uživatelské testování opravit, aby prototyp působil věrohodněji.

Na základě testování jsem se rozhodl do prototypu přidat kontextové menu transformace vyvolané po kliknutí pravým tlačítkem na blok matic. Při kliknutí pravým tlačítkem na pracovní plochu se zobrazí kompletní kontextové menu přidat (*Add*).

4.6.2 Heuristický průchod

Rozhodl jsem se podívat na prototyp z pohledu studenta. Vybral jsem si výše popsanou personu Pavla Nováka. Stránku zhodnotím podle 10 heuristických pravidel od Jacoba Nielsena, které jsou zmíněny v kapitole 4.1.

První nalezenou chybou je, že prototyp nepodporuje kroky vpřed a zpět v historii. Aktuální verze programu *I3T* tyto funkce obsahuje, bohužel v prototypu by její implementace byla časově velmi náročná a pro testování málo důležitá, proto jsem se rozhodl tuto funkcionalitu do prototypu nepřidat.

Pravidlo konzistence a standardizace porušuje ikonka ruky (na objekt lze kliknout) zobrazující se při přejíždění po pracovní ploše, která po kliknutí většinou nic nedělá. Dá se použít jen pokud je otevřené menu přidat (*add*) pro jeho zavření. Tato chyba bude v prototypu opravena a menu bude mizet po vyjetí kurzoru z menu. Bohužel v *Axure* tato funkce má menší problémy s odezvou, takže menu zmizí, až když se myší pohne od menu o větší vzdálenost.

Vybírání bloku matic je pro uživatele občas nepříjemné, protože se musí kliknout na místo v bloku matic kde není nadpis nebo matice. To dává smysl, protože to jsou pole, která se dají upravovat. Bohužel se blok nevybral ani po kliknutí na název vstupu/výstupu, kde by to mělo být možné. Tato chyba byla v prototypu opravena ještě před začátkem uživatelského testování.

Pod názvy vstupů v bloku in block a from root si uživatel nedokáže představit jaký je mezi nimi rozdíl a k čemu slouží a neexistuje žádná možnost

kde by se tuto informaci dozvěděl. V nápovědě se žádná informace ohledně těchto vstupů nenachází. Se vstupem násobení matic (*matrix*) problém není, ten je pro uživatele srozumitelný. Názvy vstupů/výstupů ponechám do uživatelského testování a v něm se uživatelů zeptám, kde by zjišťovali funkci těchto vstupů.

Úvodní text v tutoriálu, který je zobrazován stále dokola, porušuje pravidlo minimálního designu, protože je zde zobrazována zbytečná informace (úvodní text tutoriálu opakovaný na každém pod úkolu), ale během předešlých testování to žádnému uživateli nevadilo. Rozhodl jsem se zatím tento opakující text v prototypu ponechat do uživatelského testování.

Některá podmenu v menu přidat se až 3x zanořují, takže si uživatel musí zapamatovat mnoho voleb a je pro něj výběr některých položek složitý. Dokonce některá podmenu mají podobný odkaz (add–transformations a add–operator–transformations). Tento problém se dá vyřešit klávesovými zkratkami pro jednotlivá podmenu nebo přesunout některé položky z menu přidat (*add*) do vlastního menu. Během uživatelského testování viz kapitola 4.6.3 si nikdo s testovaných nestěžoval na hloubku menu.

Další nalezenou chybou byl systém propojování krabiček, kde funguje výběr jen pomocí klikání a přetáhnutí (*drag and drop*) nefunguje. Tato chyba byla objevena i v kognitivní průchodu (lepší spojování krabiček nebude z časové náročnosti do prototypu přidáno).

Chyba prototypu, která byla zjištěna i během testování prototypu kognitivním průchodem viz kapitola 4.6.1, bylo rozbalovací menu, které se chová velmi nestandardně. Bohužel to je chyba programu *Axure*, která nejde jednoduše vyřešit. V programu *I3T* fungují rozbalovací menu zcela standardně.

Jak se ukázalo i heuristický průchod provedený jedním expertem dokáže najít mnoho chyb. Některé chyby jsou sice způsobeny tím, že test probíhal jen na prototypu programu, který neobsahuje všechny funkce, ale i tak bylo nalezeno mnoho dalších chyb, které by se neměly objevit ani v prototypu.

4.6.3 Testování s uživateli

Respondenti v uživatelském testování budou plnit tutoriál na základy skládání transformací a poté jim budou zadávány další menší úkoly – smazání bloku matic a jak se po něm změní scéna, zavření okna s tutoriálem, vysvětlení některých pojmů v jednotlivých menu, jak by si představovali úvodní obrazovku, jak by třídili a hodnotili jednotlivé tutoriály, atd. Před samotným testem proběhne diskuze moderátora s testovaným o zkušenostech s podobnými programy a co by měl takový program obsahovat. Po testu proběhne diskuze se členy vývojového týmu, kteří budou vznášet dotazy na základě svých poznámek z testování.

Testy jsem moderoval osobně, protože jsem věděl o všech chybách v tutoriálu způsobených programem *Axure* a omezené funkčnosti prototypu, kterou

jsem vždy respondentům vysvětlil. Při problémech jsem tak mohl uživatelům velmi rychle pomoci s jejich řešením.

4.6.3.1 První testování s uživateli

První uživatelské testování proběhlo 2. 5. 2019 v usability lab na fakultě informačních technologií. Jeho cílem bylo otestovat celkovou podobu prototypu a zaměřit se na hlavní chyby a jejich opravy známé z předchozích testů – názvy vstupů a výstupu, připojení objektu k bloku, velkou hloubku některých menu, ignorování modré barvy zvýrazňující kroky v tutoriálu, smysluplnost textu v tutoriálu a další menší chyby. Testování se zúčastnili tři respondenti. Všichni tyto respondenti měli zkušenosti s aktuální verzí programu a při diskuzi před začátkem testování zmiňovali chyby zjištěné v analýze (např. špatné klávesové zkratky, nepřehledně označené vstupy, nevěděli o připravených scénách, atd.).

Testovaní po zvětšení křížku u objektu již neměli problém se zapojením objektu k bloku matic. Problém z názvy vstupů a výstupu zůstal, ale na základě testování bylo navrženo jeho řešení. Názvy jednotlivých vstupů/výstupů budou ponechány a bude přidána věta s popisem, která se zobrazí po najetí na vstup. S ní se zobrazí obrázek popisující funkci vstupu. Tato funkce bude implementována do prototypu do druhého testování s uživateli.

Při propojování jednotlivých bloků matic všichni uživatelé nejdříve vyzkoušeli přetažení drátu (*drag and drop*), poté jsem zasáhl a vysvětlil jim, že v programu toto fungovat bude, ale v prototypu tato funkce není implementována. Po tomto vysvětlení již uživatelé neměli problém se spojením bloků, čemuž asi hodně pomohla zkušenost s aktuální verzí programu *I3T* (prototyp i program používají symbol křížku pro výstup násobení).

Uživatelé nadále ignorovali tutoriálem modře zvýrazňovaná pole. Na konci testovaní jsem se jich na to vždy zeptal. Oni odpověděli, že si tohoto zvýraznění pořádně nevšimli, ale i tak neměli se splněným tutoriálu problémy. Nejspíše kvůli dobrým znalostem počítačové grafiky a zkušenostmi ze starší verzí programu.

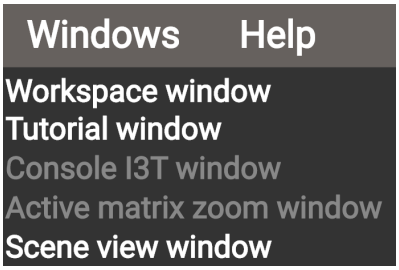
Dva ze tří testerů využili menu zobrazující se po kliknutí pravým tlačítkem do pracovní plochy. V diskuzi po testování uvedli, že jim velmi urychluje práci a že na něj jsou zvyklí z aktuální verze programu.

Třetí testovaný uživatel objevil i menu transformace (*transformations*), které se zobrazí po kliknutí pravým tlačítkem na blok matic. Očekával, že když se menu vyvolává v bloku, bude matice přidána přímo do bloku a odpadne její přesouvání z pracovní plochy do bloku matic, což se nestalo. Tuto funkci jsme se po rozhovoru s vývojovým týmem rozhodli přidat do programu, ale vzhledem časové náročnosti implementace nebude přidána do prototypu. Testovaní uživatelé, kteří toto menu využijí v prototypu budou informováni, že ve finální verzi by se matice vložila přímo do bloku.

Uživatelé při zavírání okna s tutoriálem nejdříve hledali křížek u okna, poté, když nikde nebyl, se rozhodli na okno kliknout pravým tlačítkem, ale žádné menu se nezobrazilo. Poté všem uživatelům trvalo delší dobu než našli menu oken (*Windows*) viz Obr. 4.13, kde okno s tutoriálem zavřeli. Pro zavření okna bude přidán malý křížek do hlavičky okna v prototypu a budou přidána menu pro okno, které se zobrazí po kliknutí pravým tlačítkem na okno (v některých oknech jen na hlavičku okna), kde půjde okno zavřít. Uživatelům se nelíbil způsob, kterým byla zobrazována aktivní a neaktivní okna (zašednutím textu), působil na ně, jako že toto okno nejde otevřít (šedá barva se v mnoho programech používá pro aktuálně nedostupnou funkci). Nyní budou u jednotlivých oken zaškrtávací políčka (*checkbox*), která budou znázorňovat jejich stav.

V menu pro okno tutoriál by si testovaní uživatelé přáli přidat možnost pro ukončení tutoriálu (scéna by se nezavřela) a pro přechod na stránku s výběrem jiného tutoriálu (scéna by byla zavřena s nabídkou uložení).

Testovaní uživatelé v diskuzi s moderátorem probírali podobu úvodní stránky. Zde by si představovali nejen výběr tutoriálu a filtrování podle obtížnosti tématu a hodnocení, ale také možnost otevření prázdné scény nebo nějaké předpřipravené scény (*template*), která by například měla v sobě již připravenou kameru, objekt, atd.



Obrázek 4.13: Podoba menu oken (*windows*) v prototypu během prvního kola testování.

Během testování byly objeveny chyby v menu prototypu. První z těchto chyb je, že některá podmenu jsou níže a pro uživatele je těžší na ně najet. Druhou chybou byl špatný název ve výběrovém menu (*selection*). Obě tyto chyby budou v prototypu opraveny do dalšího testování.

Druhý tester se snažil přidat objekt v menu s náhledem scény (*scene view*), protože to tak pochopil s tutoriálu. Byl velmi zaražen, když zjistil, že toto okno žádné menu nemá. Na základě této zkušenosti bude přidáno podmenu pro přidání objektu do okna s náhledem scény. Toto menu bude přidáno do prototypu do příštího testování.

Testovaní velmi dobře hodnotili nápad se změnou barvy přímo v krabičce pro objekt místo aktuálního výběru barvy v menu, který je omezený jen na 3 barvy (v analýze byla tato funkce popisována jako přidání materiálu k objektu). Kterou by si představovali jako malý obdélník s barvou umístěný vedle obrázku objektu, po kliknutí na něj by se otevřel výběr barvy objektu (*color picker*). To by ušetřilo jeden stupeň menu pro přidání objektu a zvětšilo uživatelské možnosti pro změnu barev, podle vývojového týmu by implementace do programu neměla být náročná.

Uživatelé by ocenili, kdyby po zadání špatné hodnoty během tutoriálu, byli v okně s tutoriálem informováni o chybě. Například zobrazením další nápovědy nebo zvýrazněním položky červenou barvou. Tato funkcionalita nebude do prototypu přidána, ale do programu by měla tato možnost být přidána, nemusí ji obsahovat všechny tutoriály, ale minimálně ty pro začátečníky by další nápovědu měly obsahovat.

Úpravy v provedené v prototypu můžete vidět na CD ve složce prototyp v souboru *navrh20.rp*. Testování proběhlo na verzi prototypu *narvh19.rp*.

4.6.3.2 Druhé testování s uživateli

Proběhlo 7.5. v usability lab na FIT. Testování se zúčastnili 3 respondenti, jeden z těchto respondentů neměl, žádné zkušenosti s počítačovou grafikou, druhý znal velmi málo aktuální verzi programu *I3T* a třetí respondent se již účastnil pluralistického testování. Cílem tohoto testování bylo otestovat opravu chyb zjištěných v prvním uživatelském testování, konkrétně – nápovědu k funkci vstupu pomocí obrázku, zavírání oken, funkci zaškrtávacích políček (*checkbox*) v menu oken (*Windows*) a přidávání objektu pomocí menu v okně scény (*Scene view*).

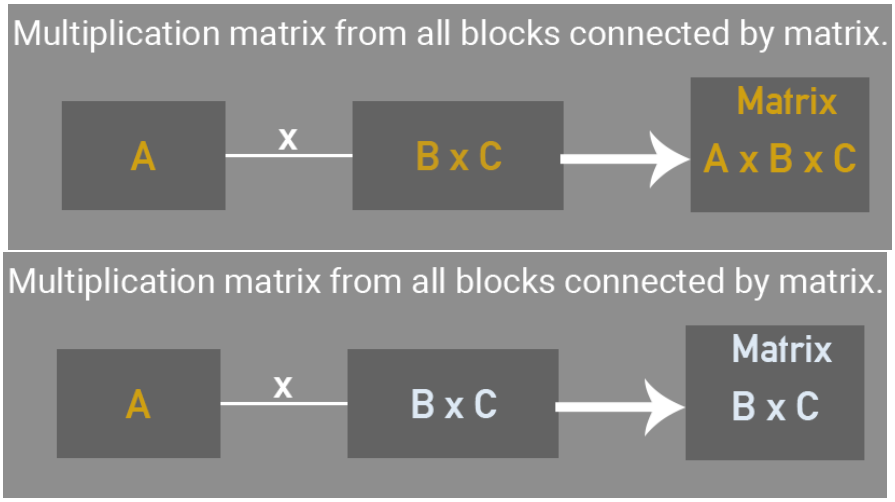
Dva ze tří testovaných vyzkoušeli přidání objektu v okně scény. Jednoho zmátlo, že zde nebyl nadpis menu a proto toto menu nevyužil. Tato chyba bude opravena, takže bude do menu přidána položka s názvem.

Všichni respondenti objevili menu „*přidat*“ (*add*), které se vyvolá po stisknutí pravého tlačítka v pracovní ploše. V diskuzi po testu říkali, že by využívali obě dvě menu „*přidat*“. Během testování uživatelé většinou používali menu, které se vyvolalo pomocí pravého tlačítka a ne menu v záhlaví pracovní plochy (*Workspace*).

Při úkolu pro smazání bloku matice, většina uživatelů zkoušela kliknout na blok matic pravým tlačítkem, bohužel v menu možnost smazání nebyla. Poté všichni uživatelé zkusili vybrat blok a smazat ho pomocí klávesy Del. Na základě testování bude do menu bloku matice přidána možnost pro smazání a duplikaci bloku (v prototypu nebudou funkční, protože by byly velmi náročné na implementaci). Nyní se v menu objevují dva typy položek, jedny položky jsou součástí podmenu transformací a druhý typ jsou volby pro okno. Tyto dva typy jsem pro větší přehlednost menu barevně odlišil viz Obr. 4.14.

S připojením objektu k bloku matic neměli testovaní žádné problémy. Testovaný bez zkušenosti s počítačovou grafikou chvíli nevěděl co má provést, poté se však podíval na objekt a hned ho připojil na výstup matrix. Takže řešení se zvětšením křížku je dostačující a uživatelům stačí.

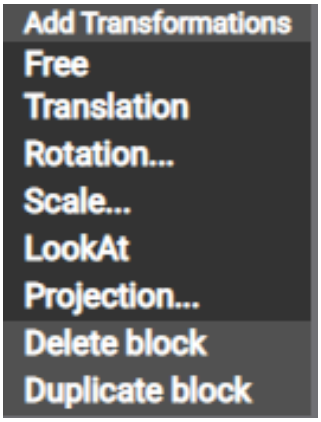
Testerů jsem se ptal na funkci jednotlivých vstupů/výstupů. S násobením neměli problém a vstupy „*in block*“ a „*from root*“ dokázali popsat díky nápovědě viz Obr. 4.15. Všichni testeři řekli, že nejvíce jim pomohl obrázek, jediné co by na obrázku změnili, by byla změna názvu operátor na matice (*matrix*), který byl ve výsledné matici u výstupu „*in block*“. Uživatele nápověda neru-



Obrázek 4.15: Na horním obrázku je nápověda pro výstup *from root* a na dolním obrázku pro výstup *in block*

šila, ale pokud by s programem pracovali déle, ocenili by kdyby se objevovala po delší době po najetí na název vstupu (v prototypu se objevuje téměř hned, bohužel se mi pro názvy nepodařilo udělat opožděné objevení (*hover*)).

Testování uživatelé uvedli, že by pro vyvolání menu matice, které by sloužilo k duplikaci nebo pro smazání, klikali na nadpis matice. Toto menu bude pro ukázku přidáno do prototypu, ale jen pouze pro matici, která se nenachází v žádném bloku a bude funkční jen možnost smazání.



Obrázek 4.14: Menu pro blok, odlišení dvou typů funkcí pomocí barvy.

Všichni tři testovaní při úkolu zavření okna použili prvně křížek. Poté jim bylo zadáno, ať okno s tutoriálem znovu otevřou. Některým uživatelům trvalo déle nalézt menu oken (*windows*) a dva dokonce vyzkoušeli položku tutoriály v menu soubor (*file*). Nakonec však všichni objevili menu oken. Bohužel lepší řešení mě ani testované nenapadlo. Ti uvedli, že po prvním použití tohoto menu, už by znovu s otevřením problém neměli.

Třetí možnost pro zavření tutoriálu (klik pravým tlačítkem na okno s tutoriálem) hledali všichni testovaní velmi dlouho, ale všichni toto menu objevili a dokázali správně popsat funkci jednotlivých tlačítek. Menší problém byl s položkou ukončit tutoriál, která zavře okna a ponechá rozpracovanou scénu,

u ní si nebyli jisti, jestli se jim neotevře obrazovka s výběrem nového tutoriálu nebo výběr nového tutoriálu v tomto okně. Po vyzkoušení této funkce, jim její podoba přišla dobrá, pouze si nebyli jisti, jestli by tuto funkci někdy využili.

Bohužel i při tomto testování uživatelé ignorovali modře zvýrazněná pole, která mají pomáhat uživatelům při plnění tutoriálu. Možná to bude tím, že zvýraznění se používá jen v některých úkolech, aby testerům nebylo moc napovídáno a uživatelské prostředí se lépe testovalo. Uživatelé toto zvýraznění přehlíželi hned u druhého úkolu a při vyplňování hodnot v matici. V diskuzi po testování ani nevěděli, že se modrá barva někde vyskytovala a napovídala jim s vyplňováním. Nevím, jestli je tento problém způsoben znalostmi uživatelů. Tuto nápovědu použil jediný testovaný uživatel v kognitivním testu, který neměl znalost matic a pomohla mu při vyplňování hodnot v maticích. Jinak uživatelé vyplňovali pole v matici na základě svých znalostí. Tuto funkci odbravení tedy nejvíce využijí úplní začátečníci, kteří nemají žádné znalosti s počítačové grafiky ani z 3D programů. Další možností je, že uživatelům tato funkce například při spojování dvou bloků napoví podvědomě a oni ji tak vůbec nevnímají. Někteří testování si mysleli, že je to chyba v prototypu a pole jen zůstalo označeno. Tato funkce by se měla znovu otestovat, až bude implementována do programu *I3T* a bude používána na všechny úkoly tutoriálu. Testování by mělo být provedeno na studentech, kteří začínají s počítačovou grafikou.

V šesté části tutoriálu bylo druhým testovaným zjištěno nevhodné spojení: „Nyní můžeme připojit blok matic k červené kostce“. Toto spojení bude v prototypu upraveno na: „Nyní připojte blok matic k červené kostce“.

Úpravy provedené na prototypu jsou v axure souboru s názvem *navrh21.rp* (lze porovnat s *navrh20.rp*), který je přiložený na CD. Na CD jsou také připojeny obrázky podoby prototypu během vývoje a vybrané axure soubory se staršími verzemi prototypu. Tento návrh (*navrh21.rp*) je finální verze prototypu. Prototyp je také dostupný na webové stránce – `https://m96fet.axshare.com`.

4.7 Shrnutí testování

V tomto odstavci shrnu chyby objevené v jednotlivých testováních. Od chyb, které se povedlo opravit po ty, které zůstávaly mezi jednotlivými testováními. Pokud jde o chybu bude u ní v závorce (P), ostatní jsou chyby návrhu uživatelského rozhraní pro *I3T*

Chyby zjištěné při pluralistickém testování:

- Názvy vstupů a výstupů sekvence a nepochopení jejich funkce
- Název sekvence – z názvu nejde poznat k čemu slouží (změna na blok matic)
- Znak pro minimalizaci krabičky

- Dlouhé texty v tutoriálu
- Propojení objektu a bloku a bloků mezi sebou

Chyby zjištěné při kognitivním průchodu:

- Názvy vstupů a výstupů bloku matic (dříve sekvence) a nepochopení jejich funkce
- Přehlížení modrého zvýraznění polí sloužícího jako nápověda v tutoriálu
- Matoucí obrázek pro 3D střed (P)
- Chyby v textu tutoriálu (P)

Chyby zjištěné heuristickou analýzou:

- Názvy vstupu a výstupů bloku matic a nepochopení jejich funkce
- Ikonka kurzoru pro kliknutí v okně s pracovní plochou (*Workspace*) (P)
- Obtížný výběr bloku matic, kliknutí pravým tlačítkem funguje jen na několika místech (P)
- Zobrazování úvodního textu ve všech krocích tutoriálu
- Některá velmi hluboká podmenu (4 vrstvy)
- Chybějící funkce historie a zkratky pro ní (jen v prototypu v programu tato funkce je) (P)

První uživatelské testování:

- Názvy vstupu a výstupů bloku matic a nepochopení jejich funkce
- Přehlížení modrého zvýraznění polí sloužícího jako nápověda v tutoriálu
- Menu transformace v bloku matic nepřidá matici přímo do bloku
- Chybějící křížek pro zavření okna a kontextové menu pro okno
- Způsob zobrazování aktivních oken v menu oken (*windows*)
- Překlepy v názvech položek v menu (P)
- Některá podmenu jsou posunuta o kousek dolu, uživatel při přejíždění na toto menu otevírá menu pod ním (P)
- Chybějící kontextové menu (pravé tlačítko) v náhledu scény (*scene view*)

Druhé uživatelské testování:

- Přehlížení modrého zvýraznění polí sloužícího jako nápověda v tutoriálu
- Chybějící název kontextového menu v okně scény (*scene view*) (P)
- Chybějící možnost pro smazání v menu pro blok matic a pro matici
- Nepřesný text v šesté položce tutoriálu (nebyl zde příkaz) (P)

4.7.1 Závěr shrnutí testování

Jak je vidět, v posledním testu jsem už mnoho chyb neobjevil a většina zde objevených chyb vznikla ve funkcích, které byli přidány v předešlém testování. Jenom ignorace modrého zvýraznění v tutoriálu zůstala nevyřešena a tento problém bude nutno otestovat na plné verzi programu *I3T*.

4.8 Pedagogické testování

Pedagogické testování probíhalo během semestru a účastnili se ho studenti počítačové grafiky na fakultě elektrotechnické a fakultě informačních technologií ČVUT v Praze. Testování vedli a vyhodnocovali učitelé Ing. Petr Felkel, Ph.D. a Ing. Jaroslav Sloup. Studenti byli rozděleni do dvou přibližně stejně velkých skupin (42 studentů pro tradiční metodu a 44 studentů pro výuku s pomocí nástroje *I3t*). Skupina *I3T* na cvičeních používala program pro výuku transformací. Jedno cvičení bylo zaměřeno čistě na transformace a zde studenti používali program *I3T* pro pochopení transformací. Druhá skupina probírala transformace standardní metodou u tabule.

Studenti nejdříve vyplnili připravený test (3. týden semestru), předtím než začali probírat transformace na přednášce. První test proběhl formou google formuláře s uzavřenými odpověďmi na cvičení. Po cvičení zabývajícím se transformacemi vyplnili tento test studenti znovu (6. týden semestru). Rozdíly mezi skupinami u tohoto testu nebyly téměř žádné (největší rozdíl byl 11 %). U některých otázek se dokonce obě skupiny zhoršily. Ve většině otázek měli lepší výsledky studenti, kteří se učili standardní formou, ale rozdíly byly jen velmi malé.

Druhý test proběhl na cvičení a psal se na papír (3. týden semestru, nejdříve se psal test formou google formuláře). Znovu se psal test dvakrát jednou před přednáškou na transformace a druhý po přednášce (6. týden semestru). Zde se studenti v šesti z osmi otázek zlepšili. U třech otázek se zlepšila úspěšnost o 20 % a více. Studenti používající při výuce *I3T* byli lepší jen ve dvou otázkách, ale o více jak 12 % . První otázka z nich byla na nakreslení levotočivého a pravotočivého souřadnicového systému a ve druhé měli studenti zjistit, zda tvoří trojice vektorů souřadnicový systém.

Studenti ze skupiny používající program pro výuku transformací naopak byli horší o více jak 10% ve třech otázkách a všechny tyto otázky se týkaly

skládání transformací a podoby jednotlivých matic. Tedy toho, na co je program zaměřen. V druhém testu je vidět větší zlepšení studentů, které může být dáno tím, že se na test více soustředí, když se píše na papír během cvičení.

Studenti po skončení testů vyplňovali dotazník a zde většina uvedla, že u aktuální verze programu *I3T* strávili 1–3 hodiny. Někteří z nich si stěžovali na uživatelské rozhraní a zmiňovali chyby, které jsou opraveny v této práci. Často by ocenili nějaký bližší popis jednotlivých funkcí, případně nějakou nápovědu při přípravě scén.

Fakt, že druhý test (post test) proběhl po přednášce a po cvičení na téma transformací neznamená, že se studenti transformace učili. Jak někteří studenti uvedli během semestru se soustředí na učení *OpenGL* a na detailnější studium transformací jim poté nezbývá čas. Při příštím testování by mohli účastníci testováni dostávat domácí úkoly (jedna skupina teoretické a druhá skupina by mohl například tvořit nějaké scény v *I3T*). Studenti by tak věnovali více času výuce transformací. Učitelé ještě nevědí, zda dají test napsat studentům ještě jednou během zkoušky, kde by studenti měli transformace umět perfektně.

Dalším problémem mohou být otázky v testu, které jsou zaměřeny hodně teoreticky. Při dalším testování by mohly test tvořit otázky, které se budou ptát postupně – nejdříve na podobu jednotlivých matic, poté jak se matice skládají a nebo co se stane s objektem, když se změní parametr v matici. Možná by tento test poskytl kvalitnější data a ukázal rozdíly mezi oběma druhy výuky.

Závěr

Cílem práce bylo seznámit se s programem pro výuku lineárních transformací, vytvořit návrh nového uživatelského prostředí programu a zanalyzovat jeho další možné funkce. Na základě analýzy ostatních programů zabývajících se lineárními transformacemi bylo vytvořeno několik návrhů, z kterých byl vytvořen prototyp programu, na kterém byly provedeny další testy a vyladěny chyby v uživatelském prostředí (např. v názvech jednotlivých položek). V novém uživatelském rozhraní by měl uživatel snadněji pochopit jednotlivé funkce programu (problematické názvy byly změněny ve spolupráci se studenty) a jeho ovládání.

Z analýzy vyplynulo, že program má velmi nestandardní zkratky, které mnoho uživatelů odradily od používání programu. Proto byl vytvořen dotazník na standardní zkratky používané v 3D programech, který vyplnilo 116 respondentů (byli zde zastoupeni i američtí studenti). Z analýzy dotazníku vznikly nové zkratky, které jsou postupně v programu nahrazovány a vyjdou v nové verzi programu. Největší problém byl se zkratkami pro ovládání 3D scény, kde se zkratky velmi lišily (každý program používá jiné).

Během semestru jsem probíral s panem Felkelem možnosti jak pedagogicky otestovat aktuální verzi *I3T*. Na základě mého připomenutí, učitelé Felkel a Sloup začali s pedagogickým testováním, v kterém rozdělili studenty na dvě skupiny. Během semestru proběhlo pedagogické testování. Z jeho dat se bohužel nedají získat žádné výsledky ohledně užitečnosti programu *I3T* při výuce lineárních transformací. V testování se možná bude pokračovat ve zkouškovém období.

Výsledný prototyp v podobě webové stránky a projektu v *Axure* bude sloužit vývojářům *I3T* při jeho reimplementaci. V prototypu zjistí, jak má fungovat funkce tutoriálu a všechna data týkající se nové podoby uživatelského prostředí (včetně grafických informací o jednotlivých položkách). Dále jsou v práci zmíněny další náměty na funkce programu, získané z diskuze se studenty, které mohou být v budoucnu do programu postupně implementovány.

Bibliografie

1. FOLTA MICHAL. *An Interactive Tool for Teaching Transformations. Version 006 (release 2018-03-22* [software]. 2019 [cit. 2019-04-17]. Dostupné z: `http://i3t-tool.org/`.
2. FLETCHER DUNN, Ian (University of North Texas; DENTON, USA) Parberry 2.vydání. *3D Math Primer for Graphics and Game Development*. 2. vydání. Taylor & Francis Inc, 2011. ISBN 1568817231. Dostupné také z: `https://www.ebook.de/de/product/13987483/fletcher_dunn_ian_university_of_north_texas_denton_usa_parberry_3d_math_primer_for_graphics_and_game_development.html`.
3. BUSS, Samuel R. *3D Computer Graphics: A Mathematical Introduction with OpenGL*. 1. vydání. Cambridge University Press, 2003. ISBN 0521821037. Dostupné také z: `https://www.ebook.de/de/product/3811944/samuel_r_buss_3d_computer_graphics.html`.
4. ANGEL, Edward; SHREINER, Dave. *Interactive Computer Graphics: A Top-Down Approach with WebGL (7th Edition)*. 7. vydání. Pearson, 2014. ISBN 0133574849.
5. 3D transformations. In: [online video] [cit. 2019-04-02]. Dostupné z: `https://www.youtube.com/watch?v=pKaxUYbl2Ok&feature=youtu.be`.
6. 3D transformations. In: [online video] [cit. 2019-04-02]. Dostupné z: `https://www.youtube.com/watch?v=rTN4nawkrZs`.
7. Geometry Transformations. *Khan Academy* [online] [cit. 2019-04-18]. Dostupné z: `https://www.khanacademy.org/math/geometry-home/transformations`.
8. Interactive 3D Graphics. *Udacity* [online] [cit. 2019-04-19]. Dostupné z: `https://eu.udacity.com/course/interactive-3d-graphics--cs291`.

9. *Open GL* [online]. Toby Rufinus et al, © 2019 [cit. 2019-04-19]. Dostupné z: https://open.gl/.

10. FOLTA, Michal. *Systém na výuku transformací*. Praha, 2016. Dostupné také z: https://dspace.cvut.cz/handle/10467/64836. Magisterská práce. České vysoké učení technické v Praze, Fakulta elektrotechnická, Katedra počítačové grafiky a interakce. Vedoucí práce Petr Felkel.

11. OpenGL transformation - release 2018-06-07. *Song Ho Ahn* [online] [cit. 2019-04-18]. Dostupné z: http://www.songho.ca/opengl/gl_transform.html.

12. AHN, Song Ho. *OpenGl ModelView Matrix* [online obrázek] [cit. 2019-04-18]. Dostupné z: http://www.songho.ca/opengl/files/gl_transform01.png.

13. CSE. *CSE 457 - Lecture demos* [software]. 2019 [cit. 2019-04-18]. Dostupné z: https://courses.cs.washington.edu/courses/cse457/03au/localdemos457.html.

14. *Geogebra* [online]. Markus Hohenwarter et al, © 2019 [cit. 2019-04-18]. Dostupné z: https://www.geogebra.org.

15. Lineární algebra a geometrické transforamace. *Veronika Havelková* [online] [cit. 2019-04-19]. Dostupné z: https://www.geogebra.org/m/QzHrupaJ.

16. PILKA, Lukáš. *I3T Úvodní obrazovka* [obrázek]. 2018 [cit. 2019-04-18] [kopie souboru uložena na přiloženém CD].

17. PILKA, Lukáš. *Návrh uživatelského rozhraní I3T* [obrázek]. 2018 [cit. 2019-04-18] [kopie souboru uložena na přiloženém CD].

18. Blender Nodes. *Blender Foundation* [online] [cit. 2019-04-25]. Dostupné z: https://docs.blender.org/manual/nb/dev/render/cycles/nodes/index.html.

19. GRAPHICS, Hyperbeam. *Nodes material editor* [online obrázek] [cit. 2019-04-25]. Dostupné z: http://hyperbeamgraphics.com/Images/cyclestest.png.

20. FOUNDATION, Blender. *Introduction to Nodes* [online obrázek] [cit. 2019-04-25]. Dostupné z: https://docs.blender.org/manual/en/latest/_images/render_blender-render_materials_nodes_introduction_nodes-default.png.

21. FOUNDATION, Blender. *Introduction to Nodes* [online obrázek] [cit. 2019-04-25]. Dostupné z: https://docs.blender.org/manual/en/latest/_images/compositing_types_distort_map-uv_example-2.png.

22. SKORUPA BARTEK, Zaal Greg. *Node Wrangler* [software]. 2019 [cit. 2019-04-26]. Dostupné z: http://gregzaal.github.io/node-wrangler/.

23. VELDHUIZEN, Bart. *Node Arrange* [software]. 2019 [cit. 2019-04-26]. Dostupné z: `https://github.com/JuhaW/NodeArrange/`.

24. EPIC GAMES. *Unreal Engine 5* [software]. 2019 [cit. 2019-04-19]. Dostupné z: `https://www.unrealengine.com/`.

25. Automatic Node Distribution and Alignment. In: *StackExchange* [online]. 2015 [cit. 2019-04-26]. Dostupné z: `https://blender.stackexchange.com/questions/41461/automatic-node-distribution-and-alignment`.

26. KRUG, Steve. *Web design - nenuťte uživatele přemýšlet!* Brno : Computer Press, 2006. 2. ISBN 80-251-1291-8.

27. NIELSEN JAKOB, Mack Robert L. *Usability Inspection Methods*. Wiley, 1994. ISBN 0-471-01877-5.

28. Heuristic Evaluation. *Usability Book of Knowledge* [online]. © 2012 [cit. 2019-04-27]. Dostupné z: `https://www.usabilitybok.org/heuristic-evaluation`.

29. Usability Book of Knowledge. *Usability Book of Knowledge* [online]. © 2012 [cit. 2019-04-27]. Dostupné z: `https://www.usabilitybok.org/`.

30. GRIGORIAN ARTEM, KOSHCHIY ALEKSANDRA, GRIGORIAN BOGDAN, DLUGOŠ MAREK. Uživatelské testování aplikace pro výuku počítačové grafiky [zpráva z testování]. 2017-12-20. [kopie souboru uložena na přiloženém CD].

31. Keyboard Shortcuts: What is more common as an redo button. In: *Quora* [online]. 2016 [cit. 2019-04-29]. Dostupné z: `https://www.quora.com/Keyboard-Shortcuts-What-is-more-common-as-an-redo-button-Ctrl-Shift-Z-or-Ctrl-Y`.

32. JINGJIE ZHENG, Daniel Vogel. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '16). In: *Finger-Aware Shortcuts*. New York, NY, USA, 4274-4285: ACM, 2016. Dostupné z DOI: `http://dx.doi.org/10.1145/2858036.2858355`.

33. AXURE SOFTWARE SOLUTIONS. *Axure 8 RP team edition. Version 8.1.0.3382* [software]. 2019 [cit. 2019-04-15]. Dostupné z: `https://www.axure.com/download`.

34. THE SIDEBAR (HTML OUTPUT). *Axure* [online] [cit. 2019-04-15]. Dostupné z: `https://www.axure.com/support/reference/sidebar-html-output`.

35. FELKEL PETR, Pilka Lukáš. I3T Tutoriál. 2018. [kopie souboru uložena na přiloženém CD].

Příloha **A**

Seznam použitých zkratek

(BI-) PGR Počítačová grafika (předmět)

(BI-) MGA Multimediální a grafické aplikace (předmět)

C++ Programovací jazyk

FEL ČVUT v Praze Fakulta elektrotechnická, České vysoké učení technické v Praze

FIT ČVUT v Praze Fakulta informačních technologií, České vysoké učení technické v Praze

GUI Grafické uživatelské rozhraní

I3T An Interactive Tool for Teaching Transformations

OpenGL Open Graphics Library

Příloha B

Scénář pluralistického průchodu

Dobrý den, vítejte na testování programu na výuku 3D transformací *I3T*. Testujeme program ne uživatele, proto se nemáte čeho bát. Testování bude probíhat na obrázcích návrhů nového vzhledu programu. Obrázky jsou dostupné na webové stránce (odkaz byl napsán na tabuli). Každý úkon bude vyobrazen pomocí obrázku, činnost kterou by jste udělali zapište vždy do položky v formuláři pro přípustnou stranu. Vždy po sekvenci kroků proběhne diskuze s vývojáři, kde nám sdělíte vaše odpovědi a to co se vám na programu nelíbí. Prosím pokud se nad něčím zastavíte poznamenejte to do nejbližší diskuze v formuláři, ať to poté můžeme probrat, případně ať si to vývojový tým může přečíst po skončení testování.

Respondenti byli informováni, aby si všímali:

- vzhledu jednotlivých oken,
- grafiky krabiček a názvů jejich výstupů a vstupů,
- zobrazení objektu v pracovní ploše (*workspace*),
- rozmístění a názvů položek v menu,
- pro vás nestandardních způsobů ovládání,
- systému několika oken,
- vzhledu okna s tutoriálem.

První obrázek je převzatý ze staršího návrhu, proto je celý v angličtině, u zbylých obrázků bude text tutoriálu v češtině, ale menu bude v angličtině, protože program je vyvíjen v angličtině. Prosím otevřete si první obrázek a dokud neřeknu nepřepínejte na další obrázek. Jednotlivé odpovědi můžete psát na papír nebo do formuláře dostupného na `http://www.i3t-tool.org/TEST/`. Vaším úkolem bude splnit tutoriál *basics transformations*.

V následujícím scénáři je vždy popis stránky a předpokládaný cíl uživatele. Dále jsou zde uvedeny časy jednotlivých diskuzí. Vzhled jednotlivých stránek najdete v příloze.

Popis jednotlivých stránek a uživatelských cílů:

První stránka bude stránka navržená Lukášem Spilkou, kde si uživatel volí jednotlivé tutoriály.

Cíl – kliknutí na tlačítko start u tutoriálu basics transformation.

Druhá stránka zde bude úvodní scéna a první úkolem bude vytvoření dalšího objektu – opice. V tutoriálu bude napsáno, že je potřeba vytvořit sekvenci.

Cíl – kliknutí na menu přidat (*Add*).

Třetí stránka Otevřené menu přidat (*add*), respondent musí vybrat položku sekvence.

Cíl – V menu přidat (*Add*) kliknout na sekvence.

Čtvrtá stránka Do pracovního okna se přidala sekvence a v tutoriálu je napsáno, že opice se přidává v menu sekvence.

Cíl – kliknutí pravým tlačítkem na sekvenci + **první diskuze**.

Pátá stránka Otevřené menu sekvence a zde na výběr podmenu.

Cíl – kliknutí na menu připnout (*bind*).

Šestá stránka Podmenu připnout (*bind*) s rovnou otevřeným custom. Přidat do sekvence opici (*monkey*).

Cíl – kliknutí na objekt opice (*monkey*).

Sedmá stránka Opice se již přidala do scény. V tutoriálu je napsáno, že uživatel musí přidat matici posunutí (*translate*).

Cíl – Otevřít menu přidat (*Add*). **Druhá diskuze** ohledně přidávání objektu do sekvence a jestli by neuvítali samostatnou krabičku objektu.

Osmá stránka Otevřené menu přidat (*add*).

Cíl – otevřít podmenu transformations.

Devátá stránka Otevřené podmenu tranformace (*transformation*).

Cíl – přidat matici posunutí (*translate*).

Desátá stránka Matice posunutí se vytvořila a teď jí chceme přiřadit k opici.

Cíl – přetažení matice do sekvence s objektem opice.

Jedenáctá stránka Posunout o jedna v ose Y, aby byla na kostce. (Pozor kostka je posunuta i v ose x). Ať popíší jednotlivé úkony a jak by změnili parametry matice.

Cíl – Změnit hodnotu x na 4 a hodnotu y na 1, jsou modře vybarveny.

Dvanáctá strana Hodnocení tutoriálu (podle návrhu 3.16), opice je správně umístěna.

Cíl – Vybrat počet hvězdiček. Nakonec proběhne **závěrečná diskuze**. Pojmenování jednotlivých vstupů a výstupů matice a jejich podoba. Zeptat se respondentů jak si myslí, že program funguje. Chtějí mít při zapnutí programu vytvořenou sekvenci nebo chtějí prázdnou scénu?

Příloha C

Text tutoriálu v prototypu

Neupravený text tutoriálu, který byl vložen do *I3T* a poté byl otestován a byly v něm opraveny chyby. Tento text se liší od textu ve finální verzi prototypu a obsahuje několik chyb. Opravený text tutoriál se používal k uživatelskému testování.

1. Dvě nejdůležitější krabičky, které můžeme přidat do scény jsou krabičky pro objekt a pro blok matic. Jako první přidejte blok matic a poté stiskněte next.

2. Do bloků matic můžeme přidávat jednotlivé matice, které se mezi sebou budou násobit. Změňte název přidaného bloku na můj první blok. Vložte do scény matici translace a přetáhněte jí do bloku. A změňte hodnotu pro osu ypsilon na 2,5 – zadejte jako 2.5 hodnoty s čárkou by (pokud by byl k bloku připojen objekt posunul by se o 2,5 dolu po ose y). (Desetinné hodnoty zadávejte s tečkou, jinak nebudou fungovat)

3. Bloky matic se mezi sebou mohou propojovat (propojení s názvem Matrix a symbolem křížku reprezentuje násobení matic). Přidejte další blok matic a změňte jeho název na posunutí v ose x.

4. Do přidaného bloku vložte matici translace a změňte její hodnotu x = 5. A tyto dva bloky propojte.

5. Cílem tohoto tutoriálu je postavit „sněhuláka“ z kostek. Za pomoci prvních dvou bloků matic bude možné se sněhulákem hýbat. Nyní budeme potřebovat další tři bloky pro každou kostku zvlášť. Vytvořte 3 bloky matic. První přidaný blok matic pojmenujte hlava, druhý tělo (blok dole) a třetí nohy (blok vpravo). Vstupy Matrix přidaných bloků napojíme na výstup Matrix bloku posunutí v ose x.

6. Nyní máme připravené jednotlivé bloky, ale potřebujeme objekty. Přidáme proto červenou kostku pro nohy sněhuláka. Kostka se objeví ve

středu scény a v pracovní ploše se objeví krabička s její reprezentací. Do bloku matic s názvem nohy přidáme matici posunutí/translace. Nyní můžeme připojit blok matic k červené kostce (kostka se posune na základě hodnot matic posunutí v jednotlivých blocích matice).

7. : Nyní si do bloku pro tělo přidáme matici translace a poté matici pro zmenšení (*scale*) a její hodnoty změníme na 0,7 (v tomto protypu se matice přidávají v pořadí přetažení do bloku, poté nejde změnit jejich pořadí a je nutné smazat blok a znovu ho přidat, dávejte proto pozor na pořadí v kterém přidáváte matice do bloku). Poté přidáme objekt zelené kostky a připojíme ji k bloku matic s názvem tělo.

8. Do scény přidáme objekt modré kostky pro hlavu, do bloku pro hlavu přidáme matice translace a zmenšení. Hodnoty pro zmenšení nastavíme na 0,5. Poté připojíme modrou kostku k bloku pro hlavu.

9. Ve scéně máme přidané tři kostky, ale vidíme jen jednu, protože kostky pro tělo a hlavu jsou menší a jsou ukryty v červené kostce. Potřebujeme proto modrou a zelenou kostku posunout v ose y a x, tak aby byly kostky nad sebou a připomínaly sněhuláka.

10. Blahopřejeme postavili jste svého prvního sněhuláka v programu I3T. Prosím ohodnoťte náš tutoriál.

Příloha D

Obsah přiloženého CD

- `readme.txt` ..... stručný popis obsahu CD
- `I3T release 2018 03 22` ..... aktuální verze programu I3T
- `I3TTUTORIALV05.docx` ..... tutoriál pro starší verzi I3T (Felkel, Pilka)
- `lukas pilka navrhy` ..... návrhy od Lukáše Pilky
- `navrhy` ..... všechny mé návrhy GUI I3T (psd i png verze)
- `prototype html` prototyp – soubory webové stránky vygenerované Axure
- `prototype snapshot` ..... obrázky jednotlivých verzí prototypu
- `src`
  - `dotaznik gify` ..... gif obrázky používané v dotazníku
  - `prototype axure` .... zdrojové soubory prototypu (axure projekty, ...)
    - `navrh21.rp` ..... finální verze prototypu
    - `stare navrhy` ..... starší axure soubory prototypu I3T
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
- `testovani`
  - `dotaznik zkratky` ..... data získaná z dotazníku na klávesové zkratky
    - `grafy` ..... grafy k datům z dotazníku
    - `odpovedi fit.png` ..... grafy odpovědí studentů FIT
  - `pluralisticke testovani` ..... podklady k pluralistickému testování
  - `uziv testovani` ..... I3T scéna sloužící jako vzor pro tutoriál
- `thesis.pdf` ..... text práce ve formátu PDF
- `zprava o testovani transformaci.pdf` . zpráva z testování A4B39TUR