České vysoké učení technické v Praze
Fakulta elektrotechnická

katedra počítačové grafiky a interakce

ZADÁNÍ DIPLOMOVÉ PRÁCE

Student: **Bc. Jaroslav Havelík**

Studijní program: Otevřená informatika (magisterský)
Obor: Počítačová grafika a interakce

Název tématu: **Rozšířená realita pro výuku**

Pokyny pro vypracování:

Nastudujte a analyzujte existující řešení (frameworky) pro rozšířenou realitu na OS Android. Vyberte vhodné řešení pro vytvoření interaktivní učebnice fyziky pro střední školy.

Vytvořte výukové scénáře respektující omezení daná vybraným frameworkem, a navrhněte a implementujte výukovou aplikaci pokrývající učební látku z elektrických obvodů s využitím prvků rozšířené reality. Finální aplikaci otestujte s reálnými uživateli. V práci zhodnoťte použitelnost vámi vytvořeného řešení pro výuku.

Témata výukových scénářů konzultujte s odborníky doporučenými vedoucím práce.

Seznam odborné literatury:

https://developer.vuforia.com/
Martin Hirzer. Marker detection for augmented reality applications. Computer graphics and vision, 2008.

Vedoucí: Ing. David Sedláček, Ph.D.

Platnost zadání: do konce letního semestru 2014/2015

L.S.

prof. Ing. Jiří Žára, CSc.
vedoucí katedry

prof. Ing. Pavel Ripka, CSc.
děkan

V Praze dne 21. 2. 2014

České vysoké učení technické v Praze
Fakulta elektrotechnická
Katedra počítačové grafiky a interakce

Diplomová práce

Rozšířená realita pro výuku

Bc. Jaroslav Havelík

Vedoucí práce: Ing. David Sedláček, Ph.D.

Studijní program: Otevřená informatika, Magisterský

Obor: Počítačová grafika a interakce

4. ledna 2015

Poděkování

Děkuji vedoucímu práce Ing. Davidu Sedláčkovi, Ph.D. za to, že mi umožnil na této práci pracovat a za velice dobrou spolupráci při tvorbě této práce. Dále bych rád poděkoval Radimu Kusákovi za mnohé konzultace a připomínky k fyzikální teorii a Davidu Urbanovi za vytvoření grafického návrhu celé aplikace. Děkuji také mým rodičům za podporu při studiu. Nakonec bych rád poděkoval mé přítelkyni za revizi tohoto textu, podporu a chvíle rozptýlení, které jsem občas potřeboval.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze dne 5. 1. 2015 .....

Abstract

This thesis focuses on the creation of mobile application for tablets, which covers high school curriculum of electrical circuits, using elements of augmented reality. Main issues of the thesis are selection of a suitable framework for augmented reality, calculation of relative position of two detected objects, generating values to the circuit, editing components of circuit and evaluation of the circuit. The application also includes a theory that covers learning material necessary for solving tasks in augmented reality. The resulting application is tested with users.

Abstrakt

Tato práce se zabývá vytvořením mobilní výukové aplikace pro tablety, která pokrývá středoškolskou látku elektrických obvodů, s využitím prvků rozšířené reality. V práci jsou analyzovány problémy jako je volba vhodné knihovny pro rozšířenou realitu, výpočet vzájemné polohy dvou detekovaných objektů, generování hodnot do obvodu, úprava jednotlivých částí obvodu a vyhodnocování elektrického obvodu. Součástí aplikace je i teorie pokrývající látku nezbytnou k řešení úloh v rozšířené realitě. Výsledná aplikace je pak otestovaná s reálnými uživateli.

Obsah

Kapitola 1

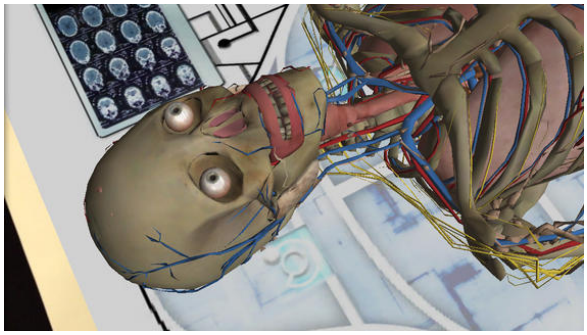
Úvod

Cílem této práce je vytvoření mobilní výukové aplikace s využitím rozšířené reality se zaměřením na učební látku fyziky (elektrické obvody) na středních školách. Rozšířená realita je kombinace mezi počítačem generovaným prostředím (virtuální realita) a reálným světem. Funguje tedy tak, že se pomocí nějakého zařízení (např. počítač, mobilní zařízení, případně zařízení připevněné k hlavě) obohacuje reálný obraz o zařízením vytvořené objekty, nejčastěji 3D modely či různé digitální prvky. Tyto objekty jsou spolu s obrazem reálného světa složeny do výsledného obrazu. Ke snímání reálného světa se využívají nejčastěji kamery zařízení (web kamery, fotoaparáty mobilních zařízení), případně jiná speciální zařízení.

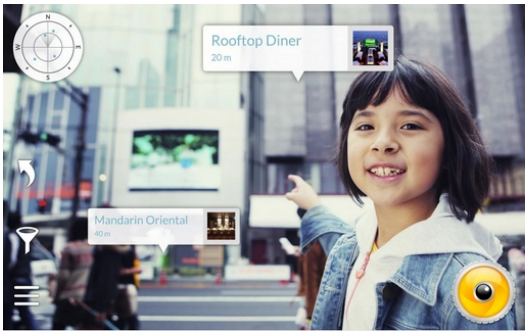
V dnešní době je rozšířená realita využívána v různých oborech, jako je například medicína, strojírenství, vzdělávání a reklama. Právě vzdělávání touto pro studenty novou formou může přinést do výuky zajímavé a interaktivní pomůcky. Vývoj mobilních zařízení jde velmi rychle vpřed a dnešní zařízení jsou natolik výkonná, že nám umožňují použití rozšířené reality v praxi při výuce. Díky velké rozšířenosti mobilních zařízení také klesá jejich pořizovací cena a stávají se dostupnější pro běžné uživatele. Základní i střední školy už začínají vybavovat učebny tablety, na kterých probíhá část výuky nejen s použitím rozšířené reality.

Příkladem využití rozšířené reality ve výuce je aplikace Anatomy 4D [8]. Pracuje na principu rozpoznávání objektů a uživatel si může na přiloženém image targetu zobrazit části lidského těla, viz obrázek 1.1, a zkoumat, jak se mezi sebou prolínají jednotlivé soustavy. Tisíce hodin strávených uživateli v aplikaci dokazují, že rozšířená realita je pro koncové uživatele velmi lákavá. Druhým příkladem je aplikace Wikitude Places [37], která pracuje s GPS modulem v mobilním zařízení. Jedná se o pomocníka při cestování, který vyhledává hotely, restaurace, obchody a bankomaty v uživatelově blízkosti. Informace o nalezených místech pak může za pomoci akcelerometru a kamery zařízení zobrazovat při namíření kamery zařízení do prostoru, viz obrázek 1.2.

Mezi dva nejčastěji používané operační systémy na mobilních zařízeních patří Android a iOS, z nichž nejrozšířenější je právě Android [32]. Jedná se o operační systém založený na jádře Linuxu, jehož výhodou je velká otevřenost a velká komunita zabývající se vývojem pro tuto platformu. Jelikož výsledkem této práce má být mobilní aplikace pro tablety s operačním systémem Android, bude tato práce vyvíjena právě pod touto platformou.



Obrázek 1.1: Anatomy 4D[1]



Obrázek 1.2: Wikitude Places[2]

1.1 Struktura tohoto textu

Následující text se skládá ze čtyř kapitol. V kapitole 2 jsou analyzovány všechny teoretické a implementační problémy spojené s touto prací a je vždy navrženo nejvhodnější řešení. Obsahem kapitoly 3 je implementace řešených problémů na základě analýzy a návrhu. V kapitole 4 jsou popsány obě fáze testování implementované aplikace. V kapitole 5 je celkové shrnutí práce, návrhy na možná vylepšení a zhodnocení práce.

[1] Převzato z <https://edshelf.com/tool/anatomy-4d#myCarousel>.
[2] Převzato z <https://play.google.com/store/apps/details?id=com.wikitude.places&hl=cs>.

Kapitola 2

Analýza a návrh řešení

V této kapitole vymezuji všechny problémy, se kterými jsem se setkal během řešení této práce. Několik prvních problémů je čistě teoretických, další jsou pak spíše praktické či implementační.

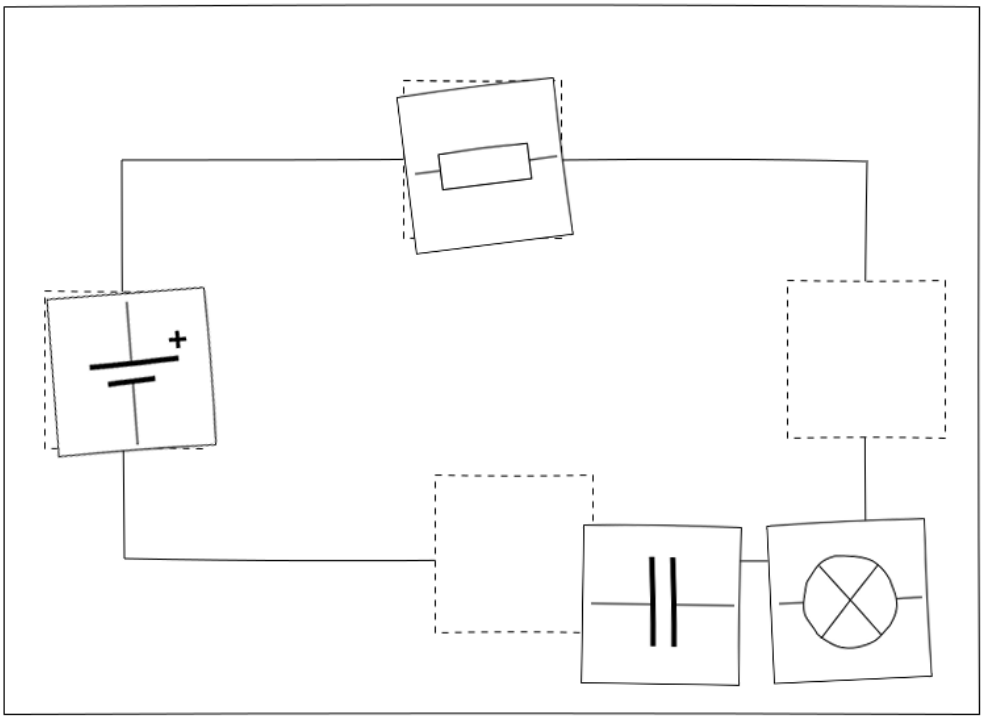
2.1 Přehled řešených problémů

Prvním analytickým problémem bylo vyřešit, jak bude fungovat plnění scénářů v rozšířené realitě. Které části obvodu budou pevné, a které bude moct student měnit. Dalším teoretickým problémem bylo zjistit, jakou látku žáci středních škol při výuce elektrických obvodů probírají, a vybrat vhodné scénáře, které by šlo nasadit na rozšířenou realitu. Dále jsem studoval jaké existují typy rozšířené reality a vymezil jsem se vůči zadání práce. Poté jsem zjišťoval, jak funguje detekce objektů v reálném světě pomocí kamery zařízení a vybral vhodnou knihovnu, která podporuje tvorbu rozšířené reality pod operačním systémem Android. Ve zvolené knihovně jsem pak porovnával možné typy objektů, které lze detekovat, a analyzoval systém, jakým je v knihovně detekce prováděna.

Mezi praktické problémy patřilo vytváření 3D modelů, které se budou zobrazovat v rozšířené realitě, a výpočet vzájemné polohy dvou objektů v rozšířené realitě. Pak také jak bude reprezentován elektrický obvod ve vnitřní paměti zařízení, jak se do něj budou generovat hodnoty, jakým způsobem se budou upravovat hodnoty vkládaných součástek a jak se bude celý obvod vyhodnocovat. Mezi poslední problémy patřil způsob uložení všech údajů o elektrickém obvodu a vytvoření teoretického základu k jednotlivým typům úloh použitých v rozšířené realitě.

2.2 Návrh konceptu řešení elektrických obvodů v rozšířené realitě

Prvním problémem bylo navrhnout vhodný systém práce s rozšířenou realitou. Zajistit, aby bylo uživateli na první pohled jasné, jak scénář splnit. Práce se scénářem měla být pro uživatele pohodlná a přitom připomínat vytváření elektrického obvodu v reálném prostředí (například ve školní fyzikální laboratoři). Dalším důležitým kritériem bylo to, aby mohl



Obrázek 2.1: Základní koncept - doplňování součástek do obvodu

být problém implementačně vyřešen pomocí dostupných knihoven podporujících rozšířenou realitou pod OS Android.

Inspiroval jsem se tedy školní fyzikální laboratoří, ve které studenti dostanou sadu součástek (spínače, žárovky, rezistory apod.) s propojovacími kabely a sestavují elektrický obvod. Využití tohoto konceptu by bylo ovšem implementačně náročné, vzhledem k nutnosti mít velké množství detekovatelných objektů (hlavně různých typů propojovacích kabelů) a také by uživatelům nemuselo být hned jasné, jak zadaný scénář splnit.

Proto jsem se rozhodl k vytvoření mírně upraveného konceptu scénáře, kde studenti dostanou síť propojovacích kabelů pevně danou (jako jednu celistvou šablonu) a pouze do ní podle zadání doplňují různé součástky (rezistory, induktory, kondenzátory). Základní idea je názorně zobrazena na obrázku 2.1. Tento koncept se ukázal jako velice blízký řešení praktických problémů, jelikož v praxi se často řeší úlohy typu jaký rezistor vložit do obvodu, aby jím protékal daný proud. Později jsem tento koncept rozšířil o možnost upravovat parametry jednotlivých součástek, viz kapitola 2.12.

2.3 Výběr vhodných scénářů pro elektrické obvody

Jelikož měla být aplikace zaměřená primárně na výuku látky elektrických obvodů na středních školách, konzultoval jsem vhodné scénáře s pedagogy ze středních škol. Prvním konzultantem byl pan Maršík (Střední průmyslová škola stavební a Obchodní akademie,

Kladno), jenž mě seznámil s látkou elektrických obvodů, která je s žáky středních škol probírána. Do této látky patří Ohmův zákon a sériové/paralelní zapojení rezistorů, Kirchhoffovy zákony, RLC obvody (obvody střídavého proudu) a elektrický proud v polovodičích. Všechny části jsou podrobně popsány v učebnici fyziky pro gymnázia – Elektřina a magnetismus [23]. Z prvních třech zmíněných okruhů vznikly postupně první testovací scénáře (ve finální aplikaci s názvy „Ohmův zákon – pokročilá úloha“, „Kirchhoffovy zákony – pokročilá úloha“ a „Obvody střídavého proudu – sériové RLC zapojení“). Poslední zmíněnou část látky (elektrický proud v polovodičích) jsem kvůli mírně jiné podstatě než u ostatních vypustil. Poté jsem připojil ještě scénář pro řešení paralelních RLC obvodů (ve finální aplikaci s názvem „Obvody střídavého proudu – paralelní RLC zapojení“).

Druhým konzultantem byl pan Kusák (Dvořákovo gymnázium a Střední odborná škola ekonomická, Kralupy nad Vltavou), který navrhl k již zmíněným testovacím scénářům přidat jejich zjednodušené varianty. Tyto scénáře měly být navrženy tak, aby procvičovaly elementární principy z právě probrané teorie. Vznikl tedy scénář s doplněním jednoho rezistoru pro procvičení Ohmova zákona, scénáře kam se doplňují dva rezistory pro procvičení sériového a paralelního zapojení rezistorů a zjednodušení scénáře pro Kirchhoffovy zákony pouze se třemi rezistory. Scénáře mají ve finální aplikaci názvy „Ohmův zákon – jednoduchý“, „Ohmův zákon – sériové zapojení rezistorů“, „Ohmův zákon – paralelní zapojení rezistorů“ a „Kirchhoffovy zákony – základní úloha“.

Výše zmíněné scénáře pokrývají většinu látky probírané na středních školách k tématu elektrické obvody.

2.4 Typy rozšířené reality podle způsobu složení výsledného obrazu

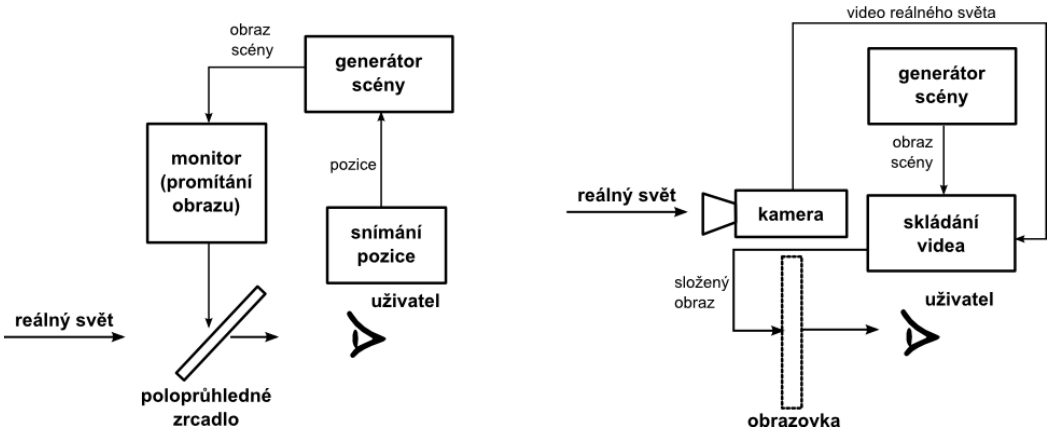
V této kapitole popíšu dvě základní rozdělení rozšířené reality podle způsobu složení výsledného obrazu, uvedu jejich výhody a nevýhody a vymezím se vůči zadání této práce.

2.4.1 Optical see throught

Prvním z typů je tzv. optical see throught [31], kde uživatel vidí reálný svět skrze poloprůhledná zrcadla umístěná před jeho oči. Zrcadla se používají k promítání přidaného obrazu před oči uživatele a dochází tak ke složení s reálným obrazem. Výhodou tohoto řešení je, že reálný svět je zobrazen absolutně nezměněný a bez žádného zpoždění. Oproti tomu velkou nevýhodou je nutnost mít na hlavě alespoň speciální zařízení, na kterém drží zmíněná zrcadla a promítá se do něj přidaný obraz, což může být v některých případech nepraktické a finančně náročné. Další nevýhodou oproti následující metodě je méně přesné mapování přidaného obrazu na obraz reálný.

2.4.2 Video see throught

Druhým způsobem je tzv. video see throught [31], kde je využito kamery zařízení, která snímá obraz reálného světa. Ten je pak digitálně složen s přidaným obrazem a zobrazen na displeji zařízení, případně promítán před oči uživatele. Výhodou této metody je přesnější



Obrázek 2.2: Porovnání Optical see throught (vlevo) a Video see trought (vpravo)

Obrázek 2.3: Jednoduchý marker s černými okraji[1]

mapování přidaného obrazu a, pokud uživateli stačí zobrazení výsledného obrazu na displeji, i to, že není nutné mít žádné další speciální pomůcky (brýle), ve kterých se obraz zobrazí. Nevýhodou je pak mírná změna reálného světa při jeho snímání čočkami kamer a mírné zpoždění zobrazení výsledného obrazu (tento problém je pomalu odbouráván stále zlepšovaným hardwarem dnešních zařízení).

Porovnání obou způsobů složení obrazu je zobrazeno na obrázku 2.2. Vzhledem k tomu, že cílem mojí práce je vytvoření mobilní aplikace bez použití dalších speciálních pomůcek, bude v této práci využit způsob video see throught.

2.5 Detekce jednoduchého markeru v rozšířené realitě

Článek [18] popisuje jednoduchý marker, viz obrázek 2.3, jako čtvercový obraz s černými okraji a vzorem uvnitř, který nese informaci o markeru. Algoritmus pro detekci markeru vyhledává čtvercové tvary s černým okrajem. Pokud nějaké najde, dopočítá pomocí čtyř

[1] Obrázek dostupný z <http://code.google.com/p/andarplus/downloads/detail?name=HIRO.jpg>.

rohů markeru homografii a odstraní perspektivní zkreslení. Poté je možné přečíst vzor uvnitř markeru a vyhodnotit, o jaký marker se jedná.

Výpočet pozice markeru a transformace ve 3D světě spočívá v odhadu transformační matice ze souřadnic markeru do souřadnic kamery. Ta je odhadnuta pomocí analýzy obrazu. Po zpracování vstupního obrazu systém hledá čtvercové oblasti a dopočítává rotační a translační komponenty transformační matice. Celý průběh výpočtu transformační matice je popsán v článku [21].

2.6 Volba knihovny pro rozšířenou realitu

V dnešní době existuje relativně velké množství knihoven (převážně komerčních), které podporují rozšířenou realitu pod OS Android. V této kapitole nejdříve uvádím několik z nich a na konci kapitoly poté vyberu nejvhodnější dle zadaných kritérií.

Mezi tato kritéria patřila hlavně podpora OS Android (jiné knihovny než s podporou OS Android níže nezmiňuji), detekce více objektů zároveň, na které lze namapovat 3D model. Posledním kritériem bylo, aby byla knihovna dostupná zdarma a s co nejmenším množstvím omezení (zobrazovaní loga knihovny, vodoznaky).

2.6.1 Layar

Layar [35] je knihovna, která slouží hlavně k rozšířené realitě v produktových katalozích firem. Funguje jako balíček aplikací, ve kterém i běžný uživatel počítače velice snadno vytvoří rozšířenou realitu. Podle dokumentace v ní lze detekovat zároveň až 50 objektů, ale není k dispozici ve volně dostupné verzi.

2.6.2 Wikitude

Wikitude [36] je velký projekt, který se zaměřuje hlavně na rozšířenou realitu s geografickou lokací, ale i snímáním markerů či reálných objektů v reálném čase. Je založen na webových technologiích (HTML, CSS, JavaScript). Výhodou je hlavně poskytnutý „Target image manager“, do kterého se nahrají obrázky pro snímání a manager vytvoří soubor snímaného objektu, se kterým knihovna pracuje. Další výhodou je „3D Encoder“, který podporuje převod 3D modelů z formátů FBX a Collada do svého formátu 3D modelu, který lze za pomoci knihovny zobrazit na detekovaném objektu. Bohužel knihovna zatím neumožňuje detekci několika objektů zároveň. Celá knihovna je dostupná v několika možných licencích, pro mé účely byla nejzajímavější licence EDU pro projekty s výukovým účelem zadarmo. Bohužel v této licenci musí být vždy vidět logo Wikitude.

2.6.3 Vuforia

Dalším velkým projektem je Vuforia [30] od společnosti Qualcomm, který se zaměřuje hlavně na rozpoznávání markerů a reálných objektů v reálném čase. Podobně jako předchozí knihovna využívá „Target manager“, do kterého lze nahrávat nejen obrázky v jedné rovině, ale i objekty složené z několika rovinných obrázků (kostka, kvádr, válec). Knihovna umožňuje

současnou detekci až pěti rovinných obrázků. Dále je možné využít tzv. frame markery, viz kapitola 2.7.2, kterých poskytuje knihovna 512, a teoreticky je možné je detekovat všechny současně. Celkově je tedy možné teoreticky detekovat až 5 + 512 objektů zároveň. Pro projekt existuje rozšíření Rajawali [19], které podporuje zobrazení 3D modelů. Toto rozšíření lze propojit pomocí dalšího projektu RajawaliVuforia [20] přímo s Vuforií. Spojením těchto knihoven dosáhneme zobrazení 3D objektu na detekovaných reálných objektech. Velkou výhodou projektu je, že celé SDK je zdarma bez žádných omezení.

2.6.4 IN2AR

IN2AR [3] je další spíše komerční produkt. Nabízí podporu detekce několika (teoreticky až 100) objektů zároveň. Má ovšem velice nepřehlednou dokumentaci a webové stránky. Navíc ve volně dostupné verzi musí být vždy zobrazeno jejich logo.

2.6.5 Metaio

Metaio [24] je velký komerční projekt, který opět podporuje detekci několika objektů zároveň. Vývoj v tomto prostředí je zadarmo, ale výsledná aplikace poté obsahuje vodoznak.

2.6.6 ARToolKit for Android

ARToolKit [1] je zástupcem open source projektů. Podporuje detekci několika markerů zároveň a dokonce i detekci reálných objektů. Knihovna podporuje práci s mobilními zařízeními s Android 2.2 a vyššími. Detekce probíhá v plném rozlišení kamery a lze využívat přední i zadní kameru zařízení. Aplikace lze psát v prostředí Java, části náročné na výkon lze psát pomocí C++ (Android NDK).

2.6.7 Zvolená knihovna

Pro přehlednost jsou všechny knihovny společně s konkrétními kritérii zobrazeny v tabulce 2.1. Podle daných kritérií nejvíce vyhovovaly knihovny Vuforia a ARToolKit. S knihovnou Vuforia jsem měl zkušenosti z předchozího studia, vyhovovala všem požadavkům a nepřinášela žádná omezení. Proto jsem se rozhodl využít právě tu společně s rozšířeními Rajawali a RajawaliVuforia.

Knihovna	Současná detekce objektů (počet)	Propojitelnost s herními a jinými enginy	Rozšířitelnost	Technologie	Cena, licence
Layar	ano (50)	žádná	uzavřené API	webové technologie	250€/měsíc
Wikitude	ne	žádná	uzavřené API	webové technologie	EDU zdarma, PRO 1499€
Vuforia	ano (5 + 512)	Unity, Rajawali	uzavřené API (nelze měnit mechanismus detekce, pouze ovlivnit, co zobrazuje)	C++, Java	zdarma, nutnost zveřejnit zdrojové kódy
IN2AR	ano (100)	Unity	uzavřené API	ActionScript3, Adobe NEL, Unity	zdarma s omezením, komerční od 999€
Metaio	ano	Unity	uzavřené API	C++, Java	zdarma vývoj, nasazení s vodoznakem
ARToolKit	ano	žádná	otevřené API	Java, C, C++	zdarma

Tabulka 2.1: Porovnání knihoven pro rozšířenou realitu

Obrázek 2.4: Porovnání nalezených významných bodů v závislosti na tvaru, významné body jsou označeny žlutým křížkem

2.7 Typy detekovatelných objektů ve zvolené knihovně a způsob jejich detekce

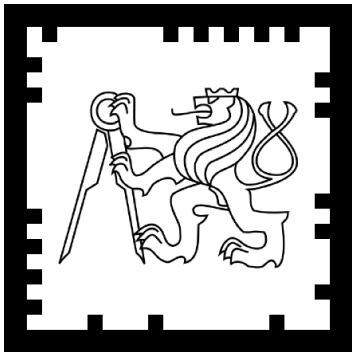
Jak jsem již zmínil v kapitole 2.6.3, zvolená knihovna poskytuje dva možné typy detekovatelných objektů, které jsou vhodné pro použití v této práci. Jedná se o tzv. image targety, kterých je možné detekovat až 5 zároveň, a frame markery, kterých je teoreticky možné detekovat až 512 zároveň. V této kapitole nejdříve popíšu, jak oba objekty vypadají a jak probíhá jejich detekce. Poté vysvětlím, jak byly tyto objekty nasazeny na základní koncept řešení scénáře z kapitoly 2.2.

2.7.1 Image target

Na rozdíl od tradičních markerů používaných v rozšířené realitě, na image targetu [29] nemusí být žádné speciální oblasti (např. černý rámeček). Pro detekci tohoto objektu jsou ve Vuforii využity tzv. významné body. Významný bod je ostrý, špičatý detail s vysokým kontrastem v obrazu. To znamená, že čtvercové tvary v obrazu poskytnou více významných bodů než oblé tvary, viz obrázek 2.4. Důležitý je také lokální kontrast, ostřejší přechod mezi dvěma barvami poskytne více významných bodů než přechod plynulý. Dále je také vhodné, aby byly významné body rovnoměrně rozprostřeny po celé ploše obrazu a aby se v obrazu neopakoval pravidelný vzor (např. šachovnice, fasáda domu).

Vhodný obraz pro image target by tedy měl být bohatý na detail (např. koláž nebo skupina lidí) a měly by se v něm střídat světlé a tmavé plochy (kontrasty). Obraz musí být ve formátu PNG (8 či 24 bit) nebo JPEG (8 bit v odstínech šedi či 24 bit v barevném prostoru RGB) a jeho velikost nesmí přesáhnout 2 MB. Obraz se nahraje do zmíněného Target Manageru, který jej vyhodnotí, nalezne v něm významné body a vytvoří dva soubory. Prvním je .xml soubor s definicemi jednotlivých targetů (unikátní jméno, velikost) a druhým .dat soubor s množinami významných bodů. Tyto soubory se přiloží do Android projektu do složky assets. Při vlastní detekci se pak aplikuje obdobný způsob jako je popsaný v kapitole 2.5.

Výhodou image targetu je, že pro úspěšnou detekci stačí, aby byla nalezena pouze podmnožina významných bodů. Nemusí být tedy celý v záběru kamery. Nevýhodou je relativně obtížné vytvoření vhodného obrazu pro kvalitní detekci a nízký počet zároveň detekovatelných objektů.



Obrázek 2.5: Frame marker (identifikační číslo 59) s logem ČVUT

2.7.2 Frame marker

Detekce za pomocí frame markeru [28] je tradiční způsob detekce objektů v rozšířené realitě. Jedinečné ID každého markeru je binárně zakódováno do rámečku, který je okolo něj. Uvnitř markeru může pak být libovolný obrázek, výsledný marker tedy vypadá jako na obrázku 2.5. Na rozdíl od image targetů nejsou frame markery vytvářeny v Target Manageru. Ve Vuforia SDK instalaci se nachází složka assets, ve které jsou připraveny šablony rámečků pro každý z 512 markerů. Ty může uživatel použít a spolu s vnitřním obrázkem vytvořit vlastní marker.

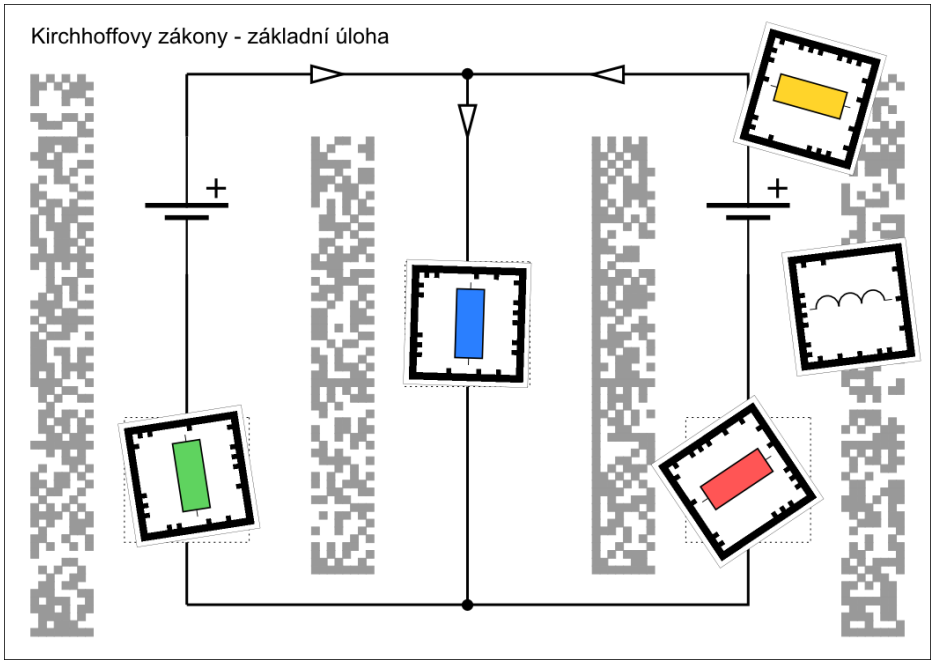
Pro detekci je pak využit způsob popsaný v kapitole 2.5. Velkou nevýhodou ovšem je, že pro úspěšnou detekci musí být celý marker v záběru kamery a nemůže být ani částečně zakrytý jiným objektem.

2.7.3 Nasazení na základní koncept

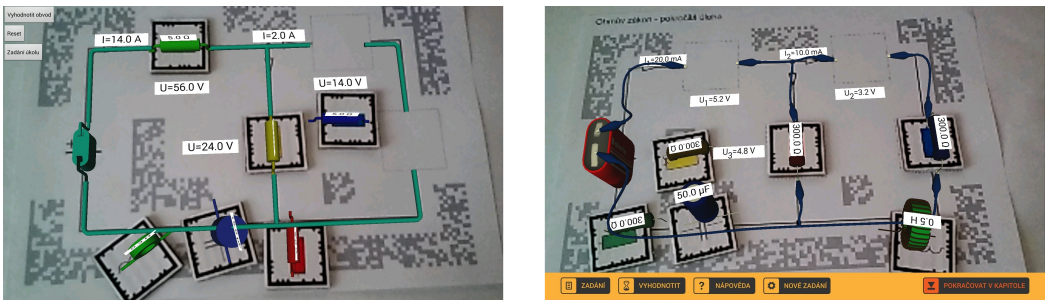
V první fázi řešení této práce jsem využíval pouze image targety jak pro pevně danou síť propojovacích kabelů, tak i pro doplňované součástky. Toto řešení velmi limitoval počet zároveň rozpoznatelných objektů (mohl jsem detekovat pouze 4 součástky + 1 síť kabelů). Proto jsem se rozhodl pro doplňované součástky použít frame markery. Díky tomu nebyl počet doplňovaných součástek omezen. Nevýhoda viditelnosti celého markeru nebyla nijak omezující vzhledem k tomu, že se markery umísťovaly dovnitř sítě kabelů a k jejich částečnému zakrytí docházelo pouze při přesunu markeru uživatelem. Veškeré nákresy sítí kabelů a součástek byly vytvořeny v programu Inkscape [34]. Pro zlepšení detekce sítě kabelů jsem do nich přidal další oblasti s významnými body pomocí funkce Barcode – Datamatrix. Vygenerovaná matice byla rozřezána a rozmístěna okolo síťě kabelů. Příklad navržené sítě s několika frame markery je na obrázku 2.6.

2.8 Tvorba vhodných 3D modelů pro rozšířenou realitu

Vzhledem ke zkušenostem z předchozího studia jsem zvolil pro 3D modely v rozšířené realitě formát Wavefront OBJ [25]. Jedná se o velmi univerzální a rozšířený formát, který je úzce spojený s formátem Material Library File (MTL), v němž jsou definovány materiály



Obrázek 2.6: Navržená síť s frame markery, úloha „Kirchhoffovy zákony - základní úloha“



Obrázek 2.7: Porovnání verzí modelů, první verze (vlevo) a finální verze (vpravo)

3D modelu. Většina programů pro tvorbu 3D grafiky podporuje export do OBJ formátu a rozšíření Rajawali podporuje jeho načítání včetně materiálů a textur.

První verze modelů byla vytvořena pomocí Autodesk Maya [2]. Jednalo se pouze o jednoduché modely bez materiálů a textur. Nicméně pro vývoj a vyzkoušení základních principů aplikace byly dostačující a první testování s uživateli, viz kapitola 4.1, ukázalo, že i tyto modely byly pro uživatele zajímavé.

Nicméně testovací modely nebyly příliš blízké realitě, a tak po konzultaci s panem Kusákem vznikla druhá verze modelů. Pro tu jsem již použil modelář Blender [4]. Hlavním důvodem bylo to, že načítání modelů pomocí rozšíření Rajawali bylo navrženo přesně pro modely vyexportované z Blenderu a komplikovanější modely vytvořené pomocí Autodesk Maya byly načítány s chybami. Finální modely podporují materiály a textury a jsou navrženy obdobně, jako kdyby studenti skládali obvod ve fyzikální laboratoři pomocí kabelů s krokosvorkami. Porovnání verzí modelů je vidět na obrázku 2.7.

2.9 Výpočet vzájemné polohy detekovaných objektů

Základním problémem vyplývajícím z konceptu uvedeném v kapitole 2.2 je zjištění, kde přesně se součástka (frame marker) vůči síti kabelů (image target) nachází. Díky výpočtu vzájemné polohy lze zjistit, jestli je místo v síti kabelů obsazené a jestli se na něm nachází správná součástka. Rozšíření RajawaliVuforia poskytuje o každém detekovaném image targetu i frame markeru polohu (vektor 3x1) a orientaci (kvaternion) v souřadném systému kamery. Z těchto hodnot lze dopočítat vzájemnou polohu dvou detekovaných objektů.

Nazveme si polohu a orientaci image targetu jako I_p a I_o (s prvky q_0, q_1, q_2, q_3), a polohu frame markeru jako F_p(vše v souřadnicích kamery). Jejich vzájemná poloha je pak spočtena následujícím postupem. Nejprve se vytvoří opačný vektor I_p^{-1} k vektoru polohy image targetu I_p podle rovnice 2.1 a normalizovaná inverze orientace image targetu I_o^{-1} podle rovnice 2.2.

$$I_p^{-1} = \begin{bmatrix} I'_{px} \\ I'_{py} \\ I'_{pz} \end{bmatrix} = I_p \cdot (-1) = \begin{bmatrix} I_{px} \\ I_{py} \\ I_{pz} \end{bmatrix} \cdot (-1) \tag{2.1}$$

$$\begin{aligned} norm &= q_0^2 + q_1^2 + q_2^2 + q_3^2 \\ norm^{-1} &= \frac{1}{norm} \\ I_o^{-1} = \begin{bmatrix} q'_0 \\ q'_1 \\ q'_2 \\ q'_3 \end{bmatrix} &= \begin{bmatrix} q_0 \cdot norm^{-1} \\ -q_1 \cdot norm^{-1} \\ -q_2 \cdot norm^{-1} \\ -q_3 \cdot norm^{-1} \end{bmatrix} \end{aligned} \tag{2.2}$$

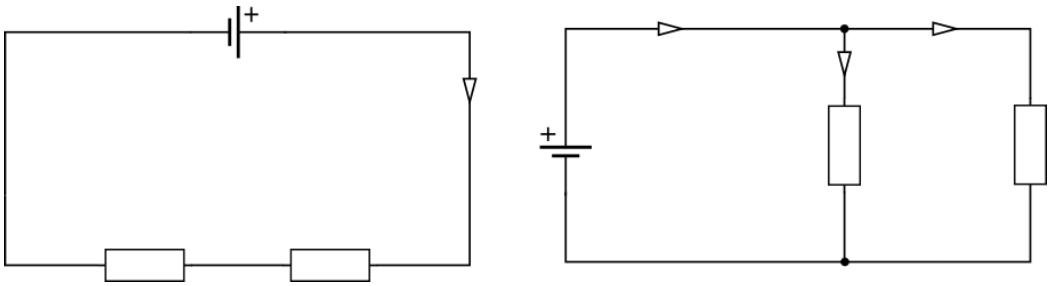
Relativní poloha závislá na orientaci kamery P_z se spočítá sečtením vektoru polohy frame markeru F_p s opačným vektorem k vektoru polohy image targetu I_p^{-1} podle rovnice 2.3. Výsledná relativní poloha P_n je spočtena vynásobením inverze orientace image targetu I_o^{-1} s relativní polohou závislou na orientaci kamery P_z pomocí rovnice 2.4.

$$P_z = \begin{bmatrix} P_{zx} \\ P_{zy} \\ P_{zz} \end{bmatrix} = F_p + I_p^{-1} = \begin{bmatrix} F_{px} + I'_{px} \\ F_{py} + I'_{py} \\ F_{pz} + I'_{pz} \end{bmatrix} \tag{2.3}$$

$$P_n = I_o^{-1} \cdot P_z = \begin{bmatrix} (1 - 2q_2'^2 - 2q_3'^2) & 2(q'_1 q'_2 + q'_0 q'_3) & 2(q'_1 q'_3 - q'_0 q'_2) \\ 2(q'_1 q'_2 - q'_0 q'_3) & (1 - 2q_1'^2 - 2q_3'^2) & 2(q'_2 q'_3 + q'_0 q'_1) \\ 2(q'_1 q'_3 + q'_0 q'_2) & 2(q'_2 q'_3 - q'_0 q'_1) & (1 - 2q_1'^2 - 2q_2'^2) \end{bmatrix} \cdot \begin{bmatrix} P_{zx} \\ P_{zy} \\ P_{zz} \end{bmatrix} \tag{2.4}$$

Při testování tohoto řešení se ovšem ukázalo, že vypočtená poloha není přesná. Proto bylo nutné do finálního algoritmu, viz kapitola 3.5, přidat ještě relaxaci v podobě konstanty ϵ, o kterou může být vypočtená poloha v každé souřadnici vyšší, respektive nižší.

Pokud tedy algoritmus vyhodnotí, že se některý z frame markerů nachází na definované pozici, přichytí model součástky na frame markeru k modelu na image targetu a nastaví pozici jako vyplněnou, viz kapitola 2.10.



Obrázek 2.8: Porovnání smyčky (vlevo) a větvení (vpravo) v obvodu

2.10 Reprezentace elektrického obvodu ve vnitřní paměti zařízení

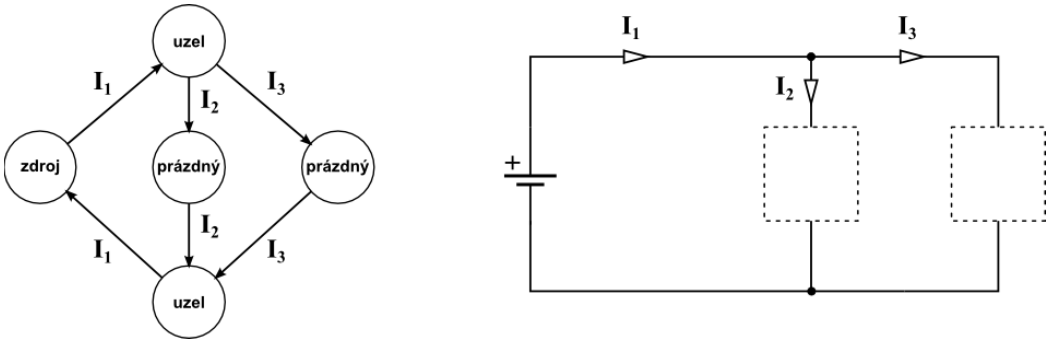
Vzhledem k tomu, že obvod měl být vyhodnocován automaticky pomocí vyhodnocovacího algoritmu aplikace, bylo nutné vytvořit vhodnou reprezentaci celého obvodu, která by se snadno procházela a ohodnocovala. Kvůli tomu, že obvod není vždy pouze jednoduchá smyčka, ale v některých případech se dělí do jednotlivých větví, viz obrázek 2.8, nebyl vhodný jednoduchý spojový seznam. Nabízelo se však použít orientovaný graf, jelikož v některých úlohách závisí na směru, kterým elektrický proud teče. Graf elektrického obvodu je spíše řídký (z každého uzlu vede malé množství hran, ve většině případů jedna nebo dvě), proto nebyla vhodná reprezentace ve vnitřní paměti pomocí matice sousednosti [6]. Zvolil jsem tedy spojovou reprezentaci grafu.

V této reprezentaci je spojový seznam všech uzlů grafu. Každý z jednotlivých uzlů ve spojovém seznamu obsahuje další spojový seznam. V mém případě všech hran, které z uzlu vycházejí. Hrana pak nese informaci o tom, ve kterém uzlu končí. Díky tomu je možné i složitější strukturu efektivně reprezentovat v paměti zařízení.

Aby mohl být obvod po doplnění vyhodnocen, musí uzel kromě seznamu hran nést i informaci o tom, zda je prázdný nebo vyplněný a o jeho elektrických veličinách (napětí, odpor, kapacita apod.). Pokud je tedy do volného místa dosazena součástka, příznak uzlu se změní z prázdného na plný a nastaví se parametry elektrických veličin podle dosazené součástky. Speciálním případem uzlu je zdroj napětí, který je v obvodu pevně vsazen a má hodnoty (napětí, v případě střídavého zdroje pak i frekvenci) vygenerované od začátku spuštění scénáře, viz kapitola 2.11, a neměnné. Druhým speciálním případem uzlu je uzel v obvodu. Jedná se o místo, kde se obvod dělí na jednotlivé větve, a nese pouze informaci o tom, které hrany z uzlu vycházejí. Hrana má pak kromě informace o tom, do kterého uzlu vede, ještě údaj o protékajícím elektrickém proudu. Příklad grafu s jeho korespondujícím obvodem je na obrázku 2.9.

2.11 Generování hodnot do elektrického obvodu a jejich zobrazení v něm

Jelikož měly jednotlivé scénáře opakovaně zkoušet studenty ze získaných vědomostí, bylo nutné vytvořit generátor hodnot pro každý z nich tak, aby pouze mechanicky nevyplňovali



Obrázek 2.9: Příklad grafu s jeho korespondujícím obvodem

již dříve známé hodnoty. Generováním hodnot také odpadl problém s opisováním výsledků od spolužáků při výuce v jedné učebně, neboť každý ze studentů měl ve svém zařízení vygenerované unikátní zadání úlohy. První možnost, která se nabízela, byla pro každý scénář vytvořit množinu zadání, ze kterých by se při spuštění úlohy náhodně jedno vybralo. Ale vytvoření této množiny by bylo příliš komplikované, a v případě rozšíření aplikace o další scénáře by bylo nutné vytvářet další množiny. Proto jsem tuto možnost zavrhl. Zvolil jsem tedy řešení, kdy se do vnitřní reprezentace obvodu generují náhodné hodnoty.

Problémem při generování hodnot bylo, že pokud by se začalo generováním hodnot proudů a napětí, které by byly poté zobrazené v zadání, počet výsledných hodnot dosazovaných součástek by byl teoreticky nekonečný a navíc by pravděpodobně docházelo k velkým zaokrouhlovacím chybám, které by mohly závažně ovlivnit finální vyhodnocování obvodu. Proto jsem ke generování hodnot musel zvolit přístup odzadu. To znamená, že se nejprve vygenerují hodnoty (např. odpor) do uzlů grafu, kam mají být doplněny elektrické součástky, a poté se dopočítají ostatní elektrické hodnoty, jako je napětí na uzlech, případně proud na hranách grafu.

Kvůli různým povahám scénářů pro rozšířenou realitu jsem musel vytvořit čtyři generátory hodnot (pro Ohmův zákon, Kirchhoffovy zákony, sériové RLC obvody a paralelní RLC obvody). Všechny generátory mají společné, že před vlastním výpočtem ostatních elektrických hodnot vygenerují hodnoty na uzly, kam se mají doplňovat elektrické součástky. Po provedení výpočtu ostatních hodnot se vygenerované hodnoty na uzlech opět vynulují (zůstanou pouze dopočtené) tak, aby byl obvod připravený pro vyhodnocování.

2.11.1 Ohmův zákon, zapojení rezistorů

Pro tento typ úlohy jsem navrhl generátor, který nejprve nalezne zdroj napětí a vygeneruje a přiřadí proud první hraně, která z něj vychází. Poté prochází celý obvod, dokud nedojde zpět ke zdroji napětí. Během celého průchodu drží hodnotu aktuálního proudu. Pokud je objeven uzel typu rezistor, z vygenerované hodnoty odporu R a aktuálního proudu I pomocí rovnice 2.5 dopočítá hodnotu napětí U. V případě, že narazí na elektrický uzel, kde se proud rozděluje do více větví, spočítá nejdříve hodnoty odporů v jednotlivých větvích a pak z hodnot aktuálního proudu I, celkového odporu ve všech větvích R_v a odporu na konkrétní větvi R_i dopočítá proud na konkrétní větvi I_i pomocí rovnice 2.6. Po dopočítání

proudu na všech větvích postupuje v každé větvi obdobně jako v předchozím případě (při objevení rezistoru spočítá hodnotu napětí).

$$U = R \cdot I \tag{2.5}$$

$$I_i = \frac{I \cdot (R_v - R_i)}{R_v} \tag{2.6}$$

2.11.2 Kirchhoffovy zákony

Generátor hodnot pro Kirchhoffovy zákony nejprve začne nalezením všech jednoduchých smyček (cyklů) v grafu. To je provedeno pomocí Tarjanova algoritmu pro nalezení elementárních cyklů [33]. Po nalezení všech cyklů se pro každý z nich vygeneruje hodnota proudu, která se přičte každé hraně v cyklu. Díky tomu je zaručen první Kirchhoffův zákon pro uzel obvodu, viz rovnice 2.7, kde proudy přicházející do uzlu mají kladné znaménko a proudy odcházející z uzlu mají záporné znaménko. Dalším krokem je průchod každého cyklu, kdy se dopočítává hodnota napětí v něm, která se pak rovnoměrně rozdělí mezi všechny zdroje nalezené v příslušném cyklu. Tím je zaručen i druhý Kirchhoffův zákon, viz rovnice 2.8, neboli součet úbytků napětí na spotřebičích (rezistorech) se v uzavřené části obvodu (smyčce) rovná součtu elektromotorických napětí zdrojů v této části obvodu.

$$\sum_{k=1}^{n} I_k = 0 \tag{2.7}$$

$$\sum_{k=1}^{n} R_k \cdot I_k = \sum_{j=1}^{m} U_{ej} \tag{2.8}$$

2.11.3 Sériové RLC obvody

Tento generátor slouží pouze pro generování hodnot do jednoduchého sériově zapojeného RLC obvodu. Ten obsahuje pouze střídavý zdroj napětí, rezistor, kondenzátor a induktor, které jsou zapojeny sériově za sebou. Generátor nejprve vygeneruje hodnotu celkového proudu I a přiřadí jí všem hranám v grafu(vzhledem k sériovému zapojení je hodnota proudu v celém obvodu stejná) a frekvenci f střídavého zdroje napětí. Poté iterativně projde každý uzel grafu a podle toho, jestli se jedná o rezistor, kondenzátor nebo induktor dopočítá hodnoty U_R, U_C a U_L podle rovnic 2.9. Následně se podle rovnice 2.10 spočítá celkové napětí U_m na zdroji a společně s frekvencí jej přiřadí zdroji napětí v grafu.

$$U_R = I_m \cdot R, \qquad U_C = \frac{I_m}{2 \cdot \pi \cdot f \cdot C}, \qquad U_L = I_m \cdot 2 \cdot \pi \cdot f \cdot L \tag{2.9}$$

$$U_m = \sqrt{U_R^2 + (U_L - U_C)^2} \tag{2.10}$$

Název	Značka	Násobek
kilo	k	$1000{=}10^3$
mili	m	$0,001{=}10^{-3}$
mikro	μ	$0,000001{=}10^{-6}$

Tabulka 2.2: Násobné jednotky použité v aplikaci

2.11.4 Paralelní RLC obvody

Generátor pro paralelní RLC obvod funguje obdobně jako generátor pro sériový RLC obvod. Jelikož se ale jedná o paralelní zapojení, je na rozdíl od sériového zapojení napětí v celém obvodu stejné a liší se proud. Vygeneruje se tedy napětí v obvodu a frekvence střídavého zdroje a pomocí rovnic 2.11 se dopočítají hodnoty I_R, I_C a I_L na jednotlivých součástkách. Z nich se pak podle rovnice 2.12 dopočítá hodnota celkového proudu I_m, která se posléze přiřadí hraně vycházející z uzlu reprezentujícího střídavý zdroj napětí.

$$I_R = \frac{U}{R}, \qquad I_C = U \cdot 2 \cdot \pi \cdot f \cdot C, \qquad I_L = \frac{U}{2 \cdot \pi \cdot f \cdot L} \tag{2.11}$$

$$I_m = \sqrt{I_R^2 + (I_C - I_L)^2} \tag{2.12}$$

2.11.5 Zobrazení hodnot v elektrickém obvodu

Pro zobrazení hodnot v elektrického obvodu jsem využil rozšířenou realitu tak, aby výsledný obvod připomínal nákres schématu se zobrazenými hodnotami. Do modelů jednotlivých scénářů jsem tedy přidal jednoduché bílé roviny, na které byly namapovány textury s textem označení, vlastní hodnotou a jednotkami (např. $U_1 = 18\,V$). Roviny jsem pak umístil na místa poblíž elementů, ke kterým patří. Dalším problémem byly řádově různé hodnoty některých veličin, například pro odpor jsou běžné hodnoty v řádu stovek, pro kapacitu kondenzátoru řádově 10^{-6}. To mohlo text na rovinách vytvářet příliš dlouhý. Proto jsem se rozhodl před vytvoření textury hodnot přidat převodník, který převáděl základní číselnou hodnotu na násobnou (dílčí) hodnotu, například hodnota „$I_1 = 0,003\,A$“ je převedena na hodnotu „$I_1 = 3\,mA$“. Tabulka 2.2 zobrazuje použité násobné jednotky v aplikaci.

2.12 Úprava hodnot jednotlivých součástek elektrického obvodu

Jelikož měly být do obvodu náhodně generovány hodnoty doplňovaných součástek, nebylo příliš vhodné mít pro každou konkrétní hodnotu vlastní součástku (frame marker). Proto jsem navrhl způsob, jakým upravit hodnotu dané součástky. Je při něm použit podobný koncept jako při doplňování součástky do obvodu. Vytvořil jsem další image target pracovně nazvaný „Editor součástek“. Do něj uživatel součástku může vložit a posléze upravit její hodnotu. Při vložení součástky je obdobně jako při vkládání součástky do scénáře vypočítávána vzájemná poloha součástky a editoru součástek, viz kapitola 2.9. Po vložení součástky do editoru je

zobrazen dialog, jestli chce uživatel opravdu upravovat danou součástku. Pokud jej potvrdí, zobrazí se obrazovka pro úpravy součástky. Jak ukázalo testování prototypu, viz kapitola 4.1, uživatelé velice často nechávali součástku v editoru součástek a po upravení a potvrzení úprav součástky systém opět detekoval součástku v editoru a nabízel uživateli dialog o možnosti znovu editovat tu samou součástku. Proto jsem se rozhodl po potvrzení úprav součástky přidat dialog, který vyzývá k vyjmutí součástky z editoru, pokud jí uživatel nechce editovat.

Hodnota každé součástky je obdobně jako ve vygenerovaném obvodu, viz kapitola 2.11.5, zobrazena pomocí rozšířené reality. Nad každou součástkou je umístěna bílá rovina, na které je zobrazena vlastní hodnota součástky s jednotkami upravená pomocí převodníku násobných jednotek.

2.13 Vyhodnocení elektrického obvodu

Poslední součástí scénáře je vlastní vyhodnocení obvodu. Je při něm nutné, aby byla detekována síť kabelů (základní šablona daného scénáře) tak, aby mohly být ohodnoceny vyplněné a nevyplněné součástky v obvodu. Prvním testem je právě zjištění, jestli jsou v obvodu nějaká místa prázdná. Je při tom využíváno principu výpočtu relativní polohy, viz kapitola 2.9. Pokud jsou v obvodu objevena prázdná místa, nemusí být dále ohodnocován a je označen jako špatně vyplněný. Prázdná místa jsou navíc označena červeným kruhem (opět se jedná o jednoduchou rovinu s texturou červeného kruhu a průhledným pozadím). Tento kruh zůstával v první verzi (prototypu) zobrazený až do opětovného vyhodnocení obvodu. Testování , viz kapitola 4.1, ovšem ukázalo, že je uživatel zmatený, pokud vyplní místo součástkou a červený kruh nezmizí. Proto jsem v další verzi navrhl skrytí kruhu pokaždé, když uživatel vyplní místo, které je kruhem označeno.

Pokud vyhodnocovaný obvod projde prvním testem, následuje druhý test. V něm se zjistí, jestli jsou použity správné typy součástek, které jsou povoleny používat ve scénáři. Například pokud je ve scénáři zapojení rezistorů použit kondenzátor, je scénář vyhodnocen jako špatně vyplněný a nemusí být dále vyhodnocován.

Poslední test závisí na typu vyhodnocovaného scénáře. Obdobně jako při generování hodnot do obvodu, viz kapitola 2.11, se podle typu scénáře spustí vyhodnocovací algoritmus, který projde graf scénáře a vyhodnotí ho jako správně či špatně vyplněný.

Při úspěšném vyhodnocení scénáře se uživateli v rozšířené realitě zobrazí chybějící mezivýsledky, které jsou vypočítány během běhu vyhodnocovacích algoritmů. Obdobně jako hodnoty vygenerované do obvodu jsou zobrazeny pomocí bílých rovin, na kterých je namapována textura s textem mezivýsledku. Hodnota je opět upravena převodníkem násobných jednotek. Ve scénářích pro Kirchhoffovy zákony jsou navíc zobrazeny obrázky smyček v obvodu, a to opět na rovinách s texturou obrázku smyčky na průhledném pozadí.

2.13.1 Ohmův zákon, zapojení rezistorů

Pro tyto úlohy jsem navrhl kontrolu pomocí prohledávání grafu scénáře do hloubky [6]. Algoritmus začne u zdroje elektrického napětí a rekurzivně prochází celý graf. V každém průchodu porovná hodnotu doplněného odporu (pokud je aktuální uzel typu rezistor, jinak pokračuje k druhému kroku) v uzlu grafu proti dopočítanému odporu z uložené hodnoty

napětí na uzlu a aktuálního proudu. K vypočtení odporu je použita upravená rovnice 2.5. Pokud se vypočtené hodnoty nerovnají, je obvod označen jako špatně vyplněný a algoritmus končí. V druhém kroku algoritmu se vyhodnocení spustí na všech uzlech, do kterých vede hrana z aktuálního uzlu s aktualizovanou hodnotou proudu.

2.13.2 Kirchhoffovy zákony

Vyhodnocení těchto úloh je navrženo velice podobně jako generování hodnot do obvodu, viz kapitola 2.11.2. Opět se v grafu nejprve naleznou jednoduché smyčky a všechny jsou postupně procházeny a ohodnocovány pomocí 2. Kirchhoffova zákona, viz rovnice 2.8. Při průchodu smyčkou je držena hodnota celkového napětí smyčky. V případě, že je při průchodu nalezen rezistor, vypočte se z hodnoty odporu a aktuálního proudu napětí na něm, viz rovnice 2.5, a přičte se k celkovému napětí smyčky. Pokud je při průchodu nalezen zdroj napětí, hodnota jeho napětí je odečtena od celkového napětí smyčky. Po průchodu celou smyčkou by měla být hodnota celkového napětí smyčky nulová. Pokud není, je obvod vyhodnocen jako špatně vyplněný.

2.13.3 Sériové RLC obvody

Podobně jako při generování této úlohy jsou procházeny všechny uzly iterativně a vypočítány hodnoty napětí (U_R, U_L a U_C) na jednotlivých elementech pomocí rovnic 2.9. Poté je vypočtena hodnota celkového napětí, viz rovnice 2.10, a porovnána s hodnotou napětí na zdroji střídavého napětí. V případě správného vyplnění obvodu jsou tyto hodnoty totožné, pokud se hodnoty liší, je obvod označen jako špatně vyplněný.

2.13.4 Paralelní RLC obvody

Stejně jako u vyhodnocování sériových RLC obvodů jsou i při vyhodnocování paralelních RLC obvodů procházeny všechny uzly. Na jednotlivých uzlech je pak vypočítán proud (I_R, I_L a I_C), viz rovnice 2.11, a z těchto hodnot je vypočtena hodnota celkového proudu I_m, viz rovnice 2.12. Ta je porovnána s hodnotou celkového proudu ve scénáři. Pokud je obvod správně vyplněný, měly by být obě hodnoty stejné. V případě špatného vyplnění scénáře se hodnoty liší.

2.14 Struktura konfiguračního souboru

Vzhledem k velkému množství scénářů a jejich parametrů bylo vhodné mít uloženou konfiguraci scénáře mimo kód aplikace v externím souboru. V první verzi jsem používal jednoduchý textový soubor, ale jak se rozšiřoval počet parametrů jednotlivých scénářů, správa i načítání textového souboru byla příliš složitá. Proto jsem se rozhodl zvolit formát XML [5], který byl přehlednější a umožnil snadnější načítání díky podpoře zpracování v programovacím jazyce Java.

V konfiguračním souboru jsou uloženy polohy pro doplnění součástek a jejich přichytávání k modelu scénáře. Dále pak struktura grafu (uzle a hrany) pro generování a vyhodnocování scénáře, možné typy používaných součástek a typ použitého algoritmu pro vygenerování

hodnot do scénáře a vyhodnocení scénáře. Dalšími elementy jsou definice (element ke kterému patří, poloha, velikost, rotace) všech textových rovin, na kterých jsou zobrazeny vygenerované hodnoty nebo mezivýsledky. Poslední elementy jsou textového charakteru. Je v nich zadání scénáře, které si uživatel může zobrazit při jeho řešení, a systém nápověd k příslušnému scénáři. Tu může uživatel využít během řešení, pokud si nepamatuje některý ze vzorců, případně si neví rady s řešením scénáře. Pro snadnější správu generovaných hodnot jsem do konfiguračního souboru přidal definice rozsahů jednotlivých veličin. Rozsah je vždy definován dolní a horní hranicí, počtem desetinných míst, na které je hodnota zaokrouhlena, a krokem (přírůstek možných hodnot). Například pokud bude mít rozsah dolní hranici 100, horní hranici 250 a krok 50, všechny možné hodnoty jsou 100, 150, 200, 250. Ukázka konfiguračního souboru je níže.

```
<?xml version="1.0" encoding="utf-8"?>
<scenario>
        <positions><!-- definice pozic pro přichytávání-->
        <position x="8.2" z="1.7" rotation="90.0" />
        </positions>
        <graph><!-- definice grafu-->
                <node id="0" type="8" value="0" generable="true" />
                <node id="1" type="2" value="0" />
                <edge id="0" source="0" target="1" current="0" />
                <edge id="1" source="1" target="0" current="0" />
        </graph>
        <parts><!-- povolené součástky-->
                <part type="1" />
        </parts>
        <evaluation><!-- typ generování a~vyhodnocování-->
                <test type="ohm" />
        </evaluation>
        <ranges><!-- rozsahy hodnot veličin-->
                <range type="resistance" low="1" high="10" decimals="0" step="1.0"/>
                <range type="voltage" low="10" high="50" decimals="1" step="1.0"/>
        </ranges>
        <extra><!-- definice rovin zobrazených v~rozšířené realitě-->
                <resultplane id ="0" type="voltage" width="4.0" height="1.0" x="5.0"
                    z="1.7" rotation="180.0" />
                <voltageplane id ="1" width="4.0" height="1.0" x="0.5" z="-4.0"
                    rotation="180.0" />
                <currentplane id ="0" width="4.0" height="1.0" x="5.0" z="-3.0"
                    rotation="180.0" />
        </extra>
        <task><!-- zadání scénáře-->
                <title>Ohmův zákon</title>
                <text edit="false" >tady se nachází HTML formátovaný text zadání</text>
        </task>
        <help><!-- množina nápověd-->
                <helplevel level="1">
                        <text>tady se nachází HTML formátovaný text nápovědy</text>
                </helplevel>
        </help>
</scenario>
```

2.15 Vytvoření teorie k elektrickým obvodům

Jelikož měla být aplikace nejen učební pomůckou ve školách, ale měla sloužit i uživatelům, kteří si jí stáhnou do svého tabletu doma a naučí se něco navíc k probírané látce, bylo vhodné k jednotlivým scénářům připojit i jejich teoretický základ. Navrhl jsem tedy teorii ke všem třem kapitolám (Ohmův zákon, Kirchhoffovy zákony a RLC obvody). Ta obsahovala převážně teoretické informace k látce, ale i několik interaktivních úloh, které měly zkoušet studenta, jestli si osvojil právě získanou teoretickou informaci. Návrh celé teorie jsem konzultoval s panem Kusákem. Níže jsou uvedeny návrhy interaktivních úloh v teorii.

Prvním návrhem interaktivního úkolu byla tzv. přetahovací úloha, kdy má student za úkol přetáhnout elementy na příslušné místo, například přiřadit značky a jednotky k příslušným elektrickým veličinám nebo složit vzorec pro Ohmův zákon. Pro pochopení Ohmova zákona jsem navrhl ještě úlohu, kde si student může vyplnit vlastní volt-ampérovou charakteristiku vodiče (případně doplnit naměřené hodnoty na laboratorním cvičení) do tabulky a volt-ampérová charakteristika je automaticky zobrazena v grafu vedle tabulky. Další interaktivní úlohy měly sloužit k pochopení zapojení rezistorů v sérii či paralelně. Student může měnit hodnoty odporu na rezistorech, případně napětí či proud a ostatní hodnoty se automaticky dopočítávají. Díky tomu je možné pozorovat závislosti veličin v daném zapojení. V kapitole o Kirchhoffových zákonech jsem navrhl jednoduchou interaktivní úlohu pro pochopení základních pojmů jako je uzel, větev nebo smyčka. V této úloze se po kliknutí na tlačítko zvýrazní příslušná oblast v elektrické síti. Poslední interaktivní úlohou bylo zadávání rovnic (například rovnice pro 1. Kirchhoffův zákon o uzlech v síti), kdy student dostane nákres elektrické sítě a musí pomocí vstupních tlačítek sestavit příslušnou rovnici.

Kapitola 3

Implementace

V první části této kapitoly se zmiňuji o použitém programovacím jazyku, vývojovém prostředí a použitých knihovnách. U použitých knihoven pak rozebírám jejich strukturu a části, které jsem nejvíce využíval v mojí práci. Před popisem hlavních implementačních problémů je nastíněna struktura celé aplikace jako Android projektu. Pak následují vlastní implementační problémy zmíněné v kapitole 2. Mezi ně patří hlavně načtení 3D modelů do scény, výpočet vzájemné polohy detekovaných objektů, reprezentace elektrického obvodu v paměti, generování, vyhodnocování obvodu a úprava jednotlivých součástek. Na konci kapitoly je popsáno načítání konfiguračního souboru a tvorba teorie k elektrickým obvodům.

3.1 Zvolený programovací jazyk, vývojové prostředí, použité knihovny

Celá práce je implementovaná ve vývojovém prostředí Eclipse IDE [9] v programovacím jazyku Java verze 8 Update 20. Do vývojového prostředí Eclipse IDE existuje plugin Android Development Tools (ADT) [11]. Pomocí něj lze stáhnout libovolné SDK pro Android a vyvíjet aplikace pod operačním systémem Android. Tato práce byla vyvíjena pod OS Android verze 4.4 Kitkat (API level 19).

Další potřebné věci pro vývoj jsou Cygwin [7] a Android NDK [14]. Cygwin je GNU kompilátor sloužící ke kompilování dynamických aplikací, jako jsou například sdílené knihovny pro Android NDK. Android NDK je potřebné pro využívání C++ API Vuforie a slouží pro vývoj částí aplikací, které jsou náročné na výkon, přímo v nativním kódu.

Dále jsem používal knihovnu Vuforia SDK ve verzi 2.8, později ve verzi 3.0 pro detekci objektů v reálném světě pomocí kamery zařízení. K načítání modelů do scény rozšířené reality je využita knihovna Rajawali a k propojení mezi knihovnami Vuforia a Rajawali slouží knihovna RajawaliVuforia. Jejich instalace je podrobně popsána na jednotlivých webových stránkách knihoven. Pro každou aplikaci založenou na Vuforia SDK platí, že je potřeba nejprve pomocí GNU kompilátoru zkompilovat nativní část aplikace a poté vytvořit výslednou aplikaci pomocí zvoleného vývojového prostředí. Pro používání knihoven Rajawali a RajawaliVuforia ve vlastní aplikaci je nutné je vložit do pracovního prostoru společně s projektem vlastní aplikace, označit je jako knihovny a ve vlastnostech projektu vlastní aplikace je pak přidat do používaných knihoven.

Poslední knihovnou, kterou jsem použil, je android_core dodaná vedoucím práce. Tato knihovna slouží k vytvoření teorie a interaktivních úloh v aplikaci. Obdobně jako Rajawali nebo RajawaliVuforia je pro její používání nutné ji vložit do pracovního prostoru společně s projektem vlastní aplikace, označit jí jako knihovnu a v projektu vlastní aplikace ji přidat do používaných knihoven. O jejím použití se detailněji zmíním v kapitole 3.11.

3.2 Popis Vuforia API, popis knihoven Rajawali a RajawaliVuforia

V této kapitole nejdříve popíšu API knihovny Vuforia a poté zmíním hlavní třídy používané z knihoven Rajawali a RajawaliVuforia.

3.2.1 Vuforia API

Vuforia má API v jazycích Java (pro OS Android) a C++ (pro iOS a Android Native), která jsou téměř totožná. Liší se pouze ve výsledném výkonu, kdy C++ se využívá pro části aplikace, které jsou náročné na výkon.

Mezi hlavní komponenty celé knihovny patří kamera, konvertor obrazu, tracker, renderer videa a databáze detekovaných objektů. Komponenta kamera se stará o zachycení snímku a jeho efektivní poslání do komponenty tracker. Konvertor obrazu převádí obraz z formátu kamery (například YUV12) do formátu vhodného pro rendering v OpenGL ES (například RGB565) a také do formátu jasu pro detekci objektů. Komponenta tracker obsahuje algoritmy počítačového vidění a zajišťuje detekci objektů. Poslední dvě zmíněné komponenty se starají o zobrazování snímků zachycených kamerou na displeji zařízení a o správu detekovatelných objektů. Jak spolu jednotlivé komponenty interagují, je zobrazeno na obrázku 3.1. Popis některých komponent jsem vynechal, jelikož nejsou podstatné pro mojí práci.

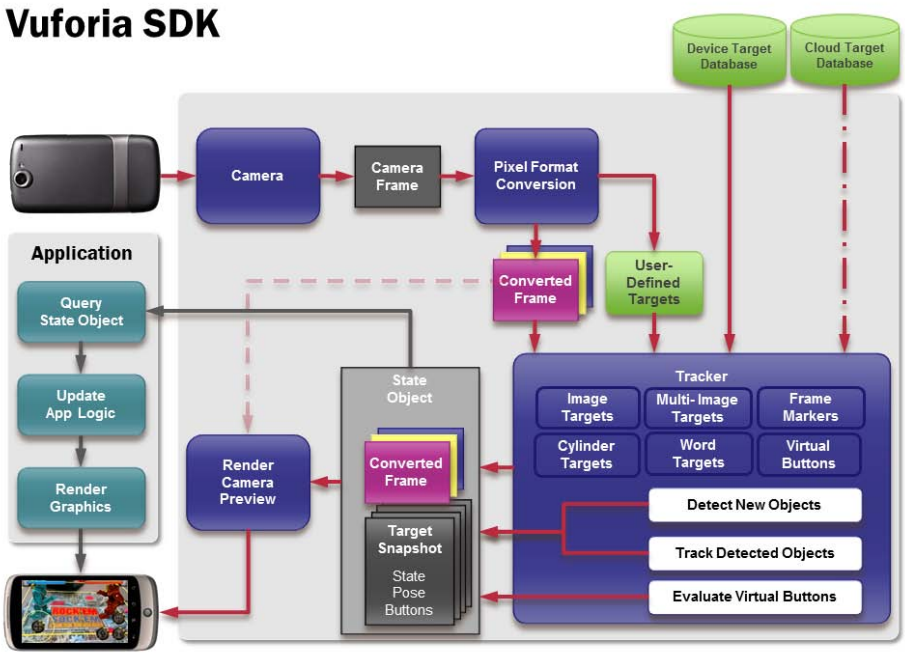
Vuforia API tedy poskytuje přístup k následujícím věcem:

- upozornění na různé události (např. dostupnost nového obrázku fotoaparátu)
- přístup k HW součástem zařízení (např. spuštění/zastavení snímání fotoaparátu)
- přístup k detekovaným objektům (image/multi targety, frame markery, ...)
- interakce s reálným světem pomocí virtuálních tlačítek

Velkým omezení Vuforia API je jeho uzavřenost. Nelze tedy měnit vnitřní algoritmy knihovny, ale pouze přistupovat k výše zmíněným událostem. Dalšími omezeními jsou například podpora pouze OpenGL ES 2.0 a Android 2.3 (Gingerbread) a vyšších.

3.2.2 Knihovna Rajawali

Nejdůležitější částí, která byla použita z této knihovny, je načítání modelů z formátu OBJ pomocí třídy LoaderOBJ. Detaily o tom, jaké vlastnosti by měl načítaný model mít, jsou v kapitole 3.4. Druhou důležitou třídou v této knihovně je třída Object3D, do které se načte 3D model z formátu OBJ. Rozšířením této třídy je třída Plane, která je v této práci



Obrázek 3.1: Vuforia SDK - jednotlivé komponenty a jejich vzájemná interakce[1]

použita pro tvorbu veškerých rovin, na kterých jsou textury s vygenerovanými hodnotami. Dalšími důležitými třídami jsou Material, Texture, AlphaTexture pro vytváření materiálů a textur.

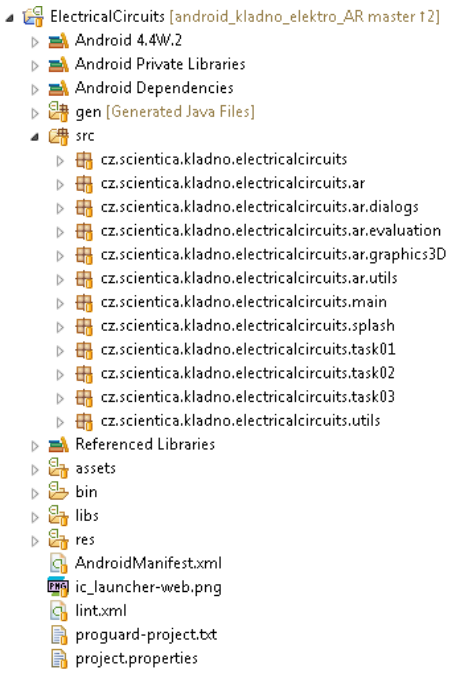
3.2.3 Knihovna RajawaliVuforia

Z této knihovny byly podstatné třídy RajawaliVuforiaActivity a RajawaliVuforiaRenderer. První z nich slouží k vytvoření a správě běhu Android aktivity. Při jejím spuštění se inicializují frame markery a image targety, připraví se uživatelské rozhraní a vytvoří renderer pro danou aktivitu. Druhá třída zajišťuje načtení modelů pomocí výše zmíněného LoaderOBJ do scény rozšířené reality, spravuje události o nalezených image targetech nebo frame markerech a případně zobrazí načtené modely na zjištěné pozici. Detaily o načtení modelů do scény a zobrazení modelu na zjištěné pozici jsou v kapitolách 3.4, respektive 3.5. Tyto třídy jsou použity zděděním do mé výsledné aplikace, kde mají názvy ARCircuitActivity respektive ARCircuitRenderer.

3.3 Popis struktury projektu výsledné aplikace

Struktura projektu výsledné aplikace vychází ze základní struktury Android projektu. Podstatné složky v projektu jsou assets, res a src. Ve složce assets jsou vloženy dva soubory, které definují vytvořené image targety, viz kapitola 2.7.1. Ve složce res jsou vloženy všechny

[1]Převzato z <https://developer.vuforia.com/resources/dev-guide/vuforia-ar-architecture>.



Obrázek 3.2: Struktura celého projektu

grafické elementy použité v aplikaci (podsložka drawable-xhdpi). Dále jsou zde vytvořeny definice rozložení jednotlivých uživatelských rozhraní (podsložka layout) a definice textových řetězců, případně použitých stylů a barev (podsložka values). Oproti běžnému Android projektu jsou v podsložce raw umístěny OBJ a MTL soubory jednotlivých modelů, konfigurační soubory všech scénářů a XML soubor sloužící pro vytvoření teorie, viz kapitola 3.11.

Složka src je rozdělena do několika balíků. Nejdůležitějším z nich je balík ar (v kořenovém adresáři cz.scientica.kladno.electricalcircuits), ve kterém jsou všechny třídy potřebné k práci s rozšířenou realitou jako ARCircuitActivity a ARCircuitRenderer zmíněné v předchozí kapitole, AREditorActivity sloužící k úpravám součástek (viz kapitola 3.8), či definice scénářů a součástek (ElectricalScenario a ElectricalComponent). Podsložkou balíku ar je balík ar.evaluation, kde se nachází všechny třídy sloužící k reprezentaci, generování a vyhodnocování elektrického obvodu. Dalšími podsložkami balíku ar jsou balíky ar.dialogs, ar.graphics3D a ar.utils. Ty definují různé dialogy zobrazované v rozšířené realitě, slouží k správě a vytváření modelů, materiálů a textur, či k načítání konfiguračního souboru a různým vlastním matematickým operacím.

Přímo v kořenovém adresáři se nachází třídy pro definice veškerých fragmentů k teorii, viz kapitola 3.11. Dále také základní aktivita (BaseActivity), od které dědí aktivity pro konkrétní kapitoly aplikace, či třída ApplicationState, ve které jsou uložena veškerá data získaná za běhu aplikace (splněné úlohy, hodnocení apod.). V balíku utils se nachází třída pro vyhodnocování textových výrazů, v balících task01, task02 a task03 jsou jednotlivé třídy pro každou ze třech kapitol aplikace. A nakonec v balíku splash a main se nachází definice úvodní obrazovky a hlavního menu celé aplikace. Celá adresářová struktura projektu je na obrázku 3.2.

3.4 Export 3D modelů a jejich načtení do scény rozšířené reality

Jak jsem již uvedl v kapitole 2.8, finální modely jsem vytvořil pomocí 3D modeláře Blender. Při exportu jsem postupoval podle doporučení uvedených na stránkách knihovny Rajawali.

Při exportu do OBJ je nutné použít následující možnosti:

- aplikovat modifikátory
- zapsat normály
- zapsat UV souřadnice
- zapsat materiály (pokud model nějaké používá)
- triangulovat plochy – LoaderOBJ podporuje pouze trojúhelníky, ne čtyřúhelníkové plochy
- objekty jako OBJ objekty

Výsledné soubory (OBJ soubor s definicí modelu a MTL soubor s definicí materiálu) mají být vloženy do složky res/raw. Vzhledem k tomu, že soubory mají kromě přípony identický název, je nutné je přejmenovat na soubor_obj.obj a soubor_mtl.mtl. Android SDK ignoruje přípony souborů, a tak by v případě nepřejmenování souborů označil soubory jako duplikáty a hlásil chybu. V případě, že model obsahuje nějaké textury, musí být umístěny v adresáři res/drawable-nodpi. Níže je uvedena ukázka kódu načtení modelu rezistor_obj ze složky res/raw a jeho přidání do scény. Tento kód lze použít ve třídě, která dědí od třídy RajawaliVuforiaRenderer, v metodě initScene.

```
try {
        // vytvori instanci parseru s~danym modelem
        LoaderOBJ objParser = new LoaderOBJ(mContext.getResources(),
            mTextureManager, R.raw.rezistor_obj);
        objParser.parse();
        rezistor = new Object3D();
        // preda zparsovany model do Object3D
        rezistor = objParser.getParsedObject();
        // prida model do sceny
        getCurrentScene().addChild(rezistor);
} catch (ParsingException e) {
        Log.i("PARSING_EXCEPTION","Model not loaded.")
}
```

Vzhledem k relativně dlouhému načítání modelů do scény (cca 30 vteřin) bylo vhodné zobrazit při načítání scény dialog, který informuje uživatele o tom, že jsou do scény načítány modely. To je dosaženo pomocí tzv. asynchronní úlohy, kde v popředí je zobrazen dialog a v pozadí se načítají jednotlivé modely. Asynchronní úloha je v Android vytvořena zděděním od třídy AsyncTask [12], ve které je nejprve pomocí metody onPreExecute vytvořen dialog, v metodě doInBackGround je prováděno vlastní načítání a v metodě onProgressUpdate je

možné upravovat dialog (například měnit text dialogu podle aktuálně načítaného modelu). Po skončení načítání je zavolána metoda onPostExecute, kde je dialog zavřen. Vlastní běh asynchronní úlohy je spuštěn pomocí metody execute.

Při inicializaci scény je tedy nejprve načten model editoru součástek a pak model vlastního scénáře. Poté jsou načteny modely doplňovaných součástek, které jsou ve scéně dvakrát, jednou jako model zobrazovaný nad frame markerem a podruhé jako součást modelu scénáře. Tak, aby se v případě přichycení součástky do scénáře zviditelnil pouze model příslušné součástky, a posunul na polohu definovanou v konfiguračním souboru, viz kapitola 2.14. Při načtení modelu součástky je také vytvořena rovina nad součástkou s hodnotou jeho odporu, kapacity nebo indukce v závislosti na typu součástky. Detaily o tom, jak je vytvořena rovina jsou uvedeny v kapitole 3.7. Po načtení všech součástek je načten model „Špatný scénář“, který se zobrazuje v případě, že uživatel namíří tablet na jinou šablonu scénáře, než se kterou má pracovat. Nakonec je do scény přidáno světlo.

3.5 Výpočet vzájemné polohy detekovaných objektů

Jak již bylo uvedeno v kapitole 2.9, vypočtení vzájemné polohy součástek (frame markerů) a šablony scénáře (image targetu) je klíčové pro vyhodnocení scénáře. Knihovna RajawaliVuforia má ve třídě RajawaliVuforiaRenderer, od které jsem ve své práci zdědil a vytvořil vlastní třídu ARCircuitRenderer, dvě metody, které oznámí nalezení frame markeru nebo image targetu. Metoda nalezení frame markeru se nazývá foundFrameMarker a obsahuje 3 parametry. Prvním z nich je identifikační číslo nalezeného markeru, druhým parametrem je pozice markeru (vektor 1×3) v souřadnicích kamery a posledním parametrem je orientace markeru (kvaternion) v souřadnicích kamery. Obdobnou strukturu má metoda nalezení image targetu foundImageMarker s tím rozdílem, že první parametr je unikátní název image targetu.

Pokud je tedy metodou foundImageMarker nalezena aktuální šablona scénáře, je zobrazen model scénáře na dané pozici a orientaci a obě hodnoty jsou uloženy. Po jejich uložení je vytvořen opačný vektor polohy scénáře a inverze orientace scénáře. Ukázka kódu uložení a vytvoření opačného vektoru a inverze kvaternionu je níže. Vše je provedeno pomocí knihovny Rajawali, která má vytvořené prostředí pro práci s vektory a kvaterniony.

```
// ulozi a inveruje pozici
this.position.setAll(position);
this.position.multiply(-1);
// ulozi a invertuje orientaci
this.orientation.setAll(orientation);
this.orientation.inverse();
```

V případě, že je metodou foundFrameMarker nalezena libovolná součástka, vypočítá se její relativní poloha vůči scénáři, viz kapitola 2.9, z opačného vektoru polohy scénáře, inverze orientace scénáře a polohy frame markeru. Celá metoda pro výpočet relativní polohy je v ukázce kódu níže.

```
/**
* Vypocita relativni pozici markeru (vzhledem k dane pozici scenare).
* @param markerPosition pozice markeru (v souradnicich kamery)
* @param scenarioPosition pozice scenare (v souradnicich kamery)
* @param scenarioOrientation orientace scenare (v souradnicich kamery)
* @return relativni pozice markeru
*/
public Vector3 getRelativePosition(Vector3 markerPosition, Vector3
    scenarioPosition, Quaternion scenarioOrientation) {
        Vector3 relative = new Vector3(markerPosition);
        relative = relative.add(scenarioPosition);
        relative = scenarioOrientation.multiply(relative);
        return relative;
}
```

Vypočtená relativní poloha je porovnávána s možnými polohami načtenými z konfiguračního souboru a v případě, že je vyhodnocena jako na poloze (s ohledem na relaxaci), zobrazí se model příslušné součástky posunutý na danou polohu a přichycený k modelu scénáře. V případě, že se součástka nenachází na žádné z možných poloh, je zobrazen její model na pozici a orientaci markeru.

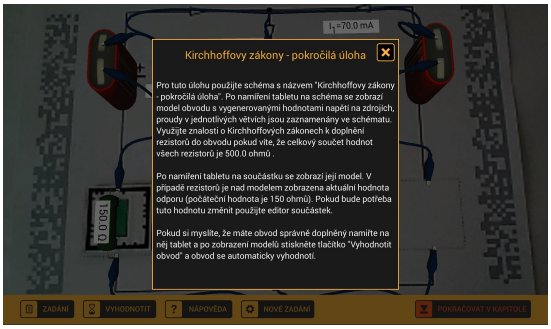
3.6 Reprezentace elektrického obvodu ve vnitřní paměti zařízení

Jádrem reprezentace elektrického obvodu je třída ElectricalScenario. V ní je definice grafu obvodu (třída CircuitGraph), ve které je spojový seznam všech uzlů grafu (třída GraphNode). Každý uzel grafu má pak jako jeden z parametrů spojový seznam hran (třída GraphEdge), které z něj vycházejí. Dalšími parametry uzlu jsou jeho identifikační číslo, příznak toho, jestli se do něj má generovat hodnota, typ uzlu (prázdný, uzel v obvodu, zdroj, rezistor apod.) a také hodnoty jednotlivých veličin (napětí, odpor, kapacita apod.). Hrana grafu má pak dva základní parametry, jedním z nich je koncový uzel hrany grafu a druhým hodnota proudu, který hranou prochází. Celá struktura grafu je načtena z konfiguračního souboru scénáře, viz kapitola 3.10.

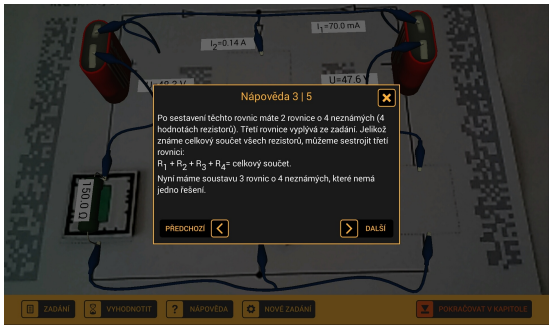
Kromě grafu obvodu je ve třídě ElectricalScenario i seznam všech doplňovaných součástek. Ten je realizován pomocí hash mapy, kde klíčem každého záznamu je identifikační číslo markeru součástky a hodnotou pak vlastní součástka. Součástky scénáře i scénář samotný implementují rozhraní Parcelable [15]. Díky tomuto rozhraní je možné složitější objekty (třídy) posílat mezi jednotlivými aktivitami pomocí jednoho příkazu. To je využito například při přechodu do aktivity pro úpravu součástek (editor součástek), viz kapitola 3.8.

Dále jsou v elektrickém obvodu seznamy názvů všech rovin (pro generování hodnot, mezivýsledky či obrázkové roviny), seznam všech poloh, na které je možno doplňovat součástky, typ používaného generování a vyhodnocení obvodu a unikátní název příslušného image targetu.

Poslední částí každého scénáře je jeho zadání a množina nápověd, které může uživatel využívat v případě, že zapomněl některý ze vzorců, případně si neví rady s řešením scénáře. Obě dvě části jsou v rozšířené realitě zobrazeny pomocí dialogu přes obrazovku rozšířené



Obrázek 3.3: Ukázka zadání úlohy



Obrázek 3.4: Ukázka nápovědy k úloze

reality, viz obrázek 3.3 a 3.4. Mezi jednotlivými stránkami nápovědy se pak uživatel může pohybovat pomocí tlačítek další a předchozí.

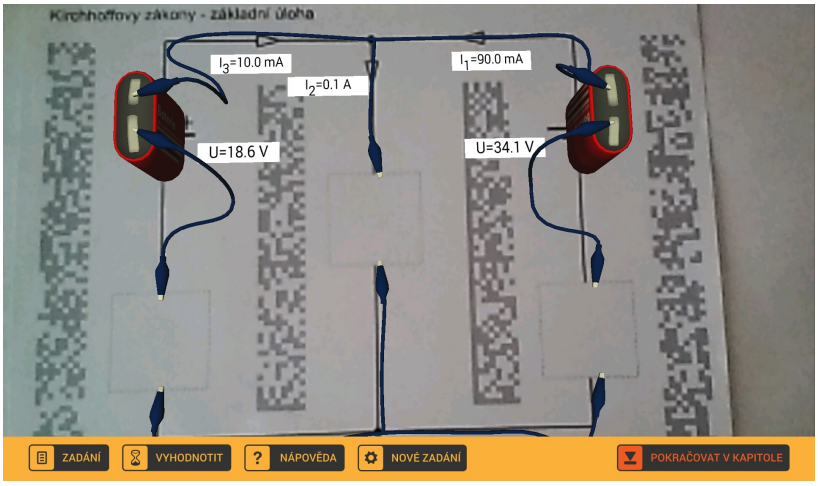
3.7 Generování hodnot do elektrického obvodu a jejich zobrazení v něm

O generování hodnot do elektrického obvodu se stará třída CircuitGenerator, která se nachází v balíku ar.evaluation. Generování hodnot se provádí pomocí statické metody generateCircuit, které se jako parametr předá příslušný scénář elektrických obvodů, do nějž jsou vygenerovány hodnoty. Z této metody se pak podle typu scénáře (Ohmův zákon, Kirchhoffovy zákony, RLC sériové, RLC paralelní) zavolá metoda příslušného generátoru.

Všechny generátory nejprve vygenerují do obvodu hodnoty doplňovaných součástek. To je provedeno v metodě generateComponents, které se parametrem předává graf příslušného elektrického obvodu. V této metodě se iterativně projdou všechny uzly a těm, které mají příznak generování nastavený na true, se vygenerují hodnoty pomocí metody generateRandomFromRange ze třídy MyMath. Této metodě se předá rozsah hodnot, ze kterého se má generovat hodnota, a vrací vygenerované číslo z rozsahu s ohledem na to, jaký má rozsah krok a na kolik desetinných míst se má zaokrouhlit. Obdobně se před ukončením generování hodnoty vygenerované pomocí metody generateComponents vynulují tak, aby byl graf scénáře připravený pro doplňování součástek při řešení scénáře a případné vyhodnocování.

Metody pro konkrétní generátory jsou nazvány generateOhm, generateKirchhoff, generateRLCSerial a generateRLCParallel. V těch je vždy vygenerován obvod podle generátorů popsaných v kapitole 2.11. Implementačně zajímavé je vyhledávání elementárních cyklů, které je převzaté z knihovny niographs [26], konkrétně ze třídy TarjanSimpleCycles. V mé práci je třída nazvaná CycleSearch a upravena pro nasazení na mojí reprezentaci grafu popsanou v kapitole 3.6.

Jak bylo uvedeno v kapitole 2.11.5, hodnoty jsou zobrazovány pomocí rozšířené reality. Na místa definovaná v konfiguračním souboru jsou pomocí třídy ModelHandler vytvořeny do 3D modelu scénáře všechny roviny s texty. Rovina je vytvořena pomocí třídy Plane v knihovně Rajawali. Vytvoření roviny s namapovanou texturou, na které je požadovaný text, je provedeno pomocí metody createTextPlane ve třídě BitmapPlaneFactory, které se jako parametry předají rozměry roviny, její text, velikost fontu a barva textu. Textura je

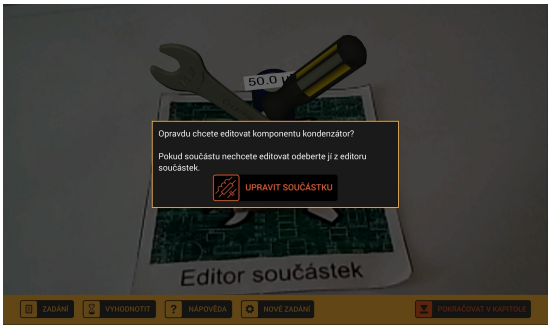


Obrázek 3.5: Obvod s vygenerovanými hodnotami

vytvářena programově jako bitmapa, do které je pomocí tříd Canvas [13] a StaticLayout [16] umístěn text. Hodnota v umístěném textu je před jeho vytvořením upravena převodníkem násobných hodnot pomocí metody formatValue ze třídy MyMath, která přijme jako parametr číselnou hodnotu a vrátí upravenou textovou hodnotu. Umístěný text také podporuje HTML formátování, takže je možné vytvářet texty s dolními indexy pro snadnější rozlišení veličin. Ukázka kódu vytvoření bitmapy s umístěným textem je zobrazena níže. Jak vypadá model elektrického obvodu s vygenerovanými hodnotami je zobrazeno na obrázku 3.5.

```
// vytvori bitmapu
Bitmap image = Bitmap.createBitmap((int) width, (int) height,
    Bitmap.Config.ARGB_8888);
// vytvori canvas a pripoji ho k bitmape
Canvas canvas = new Canvas(image);
canvas.drawColor(Color.WHITE);
// vytvori text paint s atributy (velikost a barva pisma)
TextPaint mTextPaint = new TextPaint();
mTextPaint.setTextSize(fontSize);
mTextPaint.setColor(color);
// vytvori text layout a vykresli do nej na stred formatovany text
StaticLayout mTextLayout = new StaticLayout(Html.fromHtml(text), mTextPaint,
    canvas.getWidth(), Alignment.ALIGN_CENTER, 1.0f, 0.0f, false);
// vykresli text layout do canvasu
canvas.save();
mTextLayout.draw(canvas);
canvas.restore();
// v image je nyni vysledna bitmapa
```

Při vytváření některých scénářů se ukázalo, že několik hodnot (například součet hodnot všech rezistorů v Kirchhoffových zákonech, případně napětí na jednotlivých součástkách v sériových RLC obvodech) není vhodné zobrazit pomocí rozšířené reality, jelikož nešly z jejich podstaty umístit do 3D modelu elektrického obvodu, případně nebylo před doplněním obvodu známé jejich umístění. Proto jsem se rozhodl tyto hodnoty uložit v reprezentaci scénáře (třída ElectricalScenario) a zobrazit v zadání scénáře, viz kapitola 2.14.



Obrázek 3.6: Součástka v editoru součástek s dialogem

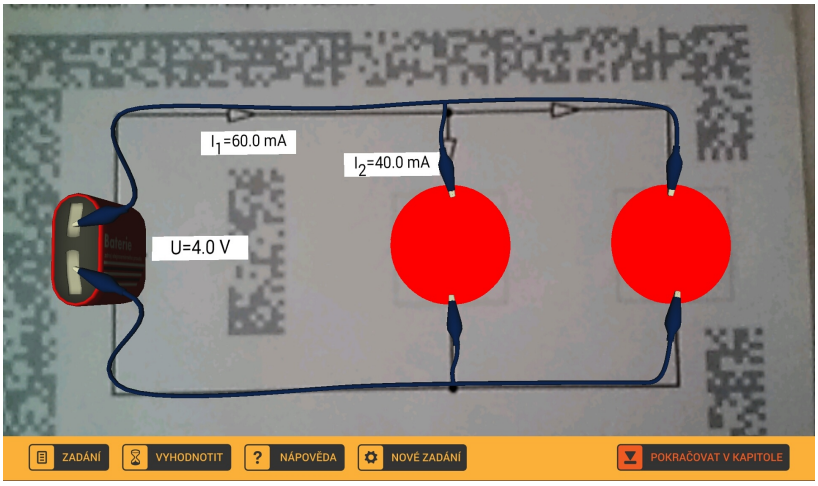
Obrázek 3.7: Ukázka obrazovky editoru součástek

3.8 Úprava hodnot jednotlivých součástek elektrického obvodu

Úprava součástek je prováděna pomocí editoru součástek (image targetu), na který uživatel vloží požadovanou součástku a poté jí může upravit. Je zde využito podobného principu jako při doplňování součástek do obvodu. Pokud je detekován image target editoru součástek, tak se počítá jeho vzájemná poloha s polohou všech detekovaných součástek (frame marker). Pokud je relativní poloha blízká nule (opět s ohledem na relaxaci hodnot), je součástka vyhodnocena jako v editoru součástek a zobrazí se dialog, jestli chce uživatel opravdu editovat součástku, viz obrázek 3.6. Pokud uživatel nechce součástku upravovat, odebere jí z editoru a dialog automaticky zmizí.

Po potvrzení dialogu je spuštěna aktivita AREditorActivity, které se předá identifikační číslo markeru upravované součástky a scénář s uloženými součástkami. Pomocí těchto parametrů je v aktivitě vyplněn název součástky, její aktuální hodnota a určen rozsah možných hodnot, které může součástka mít. Uživatel si pak může pomocí tlačítek vybrat požadovanou hodnotu součástky. Vzhled celého editoru součástek je zobrazen na obrázku 3.7. Pro potvrzení změn uživatel klikne na tlačítko „Uložit součástku a zavřít“ a je zavolána metoda editComponent. V té se nejprve nastaví součástce příslušná hodnota a poté připraví Intent pro navrácení do ARCircuitActivity. Před návratem do hlavní aktivity se ještě zobrazí dialog s upozorněním, že by měl uživatel vyjmout součástku z editoru součástek, pokud jí nechce znovu editovat. V tomto dialogu je umožněno zaškrtnout možnost „Rozumím, příště již nezobrazovat“, a při zaškrtnutí této možnosti se do SharedPreferences aplikace uloží příznak, že tento dialog už nebude nikdy zobrazen. Po potvrzení dialogu se vyvolá hlavní aktivita se změněným Intentem a je zobrazena zpráva ve formě Toastu, že byla upravena součástka na novou hodnotu.

Po návratu do hlavní aktivity je upravování zachyceno metodě onNewIntent (díky změněnému Intentu). Poté se upraví hodnoty součástky v seznamu doplňovaných součástek scénáře. Nakonec se upraví textura roviny, která se nachází nad 3D modelem součástky. Textura je vytvořena obdobně jako je popsáno v kapitole 3.7, přiřazena novému materiálu a ten je pomocí metody setMaterial přiřazen rovině.



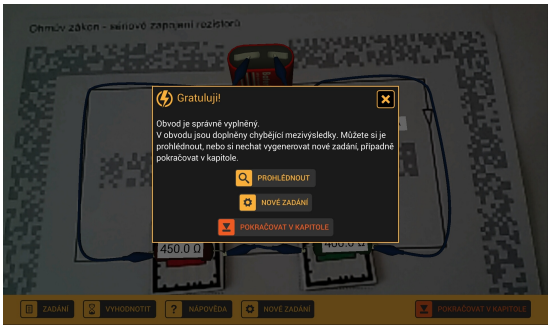
Obrázek 3.8: Vyhodnocený obvod s prázdnými místy

3.9 Vyhodnocení elektrického obvodu

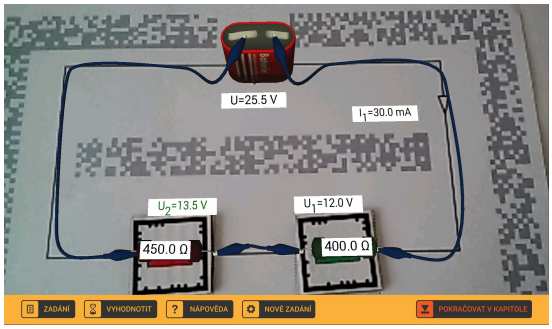
Vyhodnocení elektrického obvodu je prováděno v třídě CircuitGraph, konkrétně pomocí metody evaluate. Ta nejprve zkontroluje, jestli obvod obsahuje nějaká prázdná místa, případně je označí červeným kruhem viz obrázek 3.8. Pokud jsou všechna místa v obvodu zaplněná, zkontroluje jestli jsou doplněné součástky správného typu, a poté spustí vyhodnocení podle typu scénáře.

Vyhodnocení podle konkrétního typu scénáře je naimplementováno tak, jak je popsáno v kapitole 2.13. Jednotlivé metody pro vyhodnocení jsou correctOhmLaw, correctKirchhoffLaw, correctRLCSerial a correctRLCParallel. V každém kroku vyhodnocovacích algoritmů jsou ukládány mezivýsledky hodnot do HashMapy, kde klíčem každého záznamu je identifikační číslo uzlu nebo hrany, ke kterému patří, a hodnotou je instance třídy ResultType. Ta obsahuje typ mezivýsledku (napětí nebo proud) a jeho vlastní hodnotu. Aby nedocházelo ke kolizím identifikačních čísel uzlů a hran, je k identifikačnímu číslu hrany přičtena vysoká konstanta. V případě, že je obvod vyhodnocen jako chybně vyplněný, zobrazí se zpráva ve formě Toastu, která říká, co je příčinou špatného vyhodnocení (prázdná místa v obvodu, použití špatných součástek nebo špatné hodnoty doplňovaných součástek).

Při úspěšném vyhodnocení elektrického obvodu je zobrazen dialog, který nabízí možnost prohlédnout si obvod s doplněnými mezivýsledky, pokračovat v kapitole nebo si nechat vygenerovat nové zadání a vyzkoušet si scénář splnit znovu. Dialog je zobrazen na obrázku 3.9. V případě, že se uživatel rozhodne prohlédnout si obvod s mezivýsledky je zavolána metoda displayResults třídy ModelHandler. Ta postupně vytvoří materiály pro načtené roviny mezivýsledků obdobně jako je vytvořen materiál na roviny v kapitole 3.7. Roviny s přiřazenými materiály poté zviditelní v 3D modelu scénáře. V případě Kirchhoffových zákonů navíc zobrazí obrázky dvou smyček v obvodu. Příklad výsledného 3D modelu správně vyhodnoceného elektrického obvodu je na obrázku 3.10.



Obrázek 3.9: Dialog „Gratuluji“ po správném vyhodnocení obvodu



Obrázek 3.10: Správně vyplněný obvod s mezivýsledkem

3.10 Načtení konfiguračního souboru

Jak již bylo zmíněno v kapitole 2.14, každý scénář má uloženou svojí konfiguraci v externím souboru ve formátu XML. K načítání souboru je využit DOM parser [27]. Načtení souboru zajišťuje třída ConfigParser, která se nachází v balíku ar.utils. Hlavní metodou této třídy je statická metoda parseConfig, jejíž jediným parametrem je scénář, do kterého se mají načíst hodnoty z konfiguračního souboru. Ten je uložen ve složce res/raw. V metodě parseConfig se tento soubor otevře a v dalších metodách volaných z hlavní metody se postupně načítají jednotlivé části konfiguračního souboru. Ukázka kódu otevření souboru je zobrazena níže.

```
// vytvori factory a builder
DocumentBuilderFactory dbFactory = DocumentBuilderFactory.newInstance();
DocumentBuilder dBuilder;
dBuilder = dbFactory.newDocumentBuilder();
// zparsuje a normalizuje soubor
org.w3c.dom.Document doc = dBuilder.parse(scenario.context.getResources()
    .openRawResource(id_souboru));
doc.getDocumentElement().normalize();
```

Jednotlivé části se pak načítají téměř stejným postupem. Nejprve se v souboru naleznou všechny elementy s požadovaným názvem a poté se jeden po druhém načítají společně s atributy elementu. Ukázka kódu nalezení všech požadovaných elementů a zpracování jejich atributů je zobrazena níže. Výjimkou jsou elementy pro zadání a nápovědu ke scénáři, kde se nachází text zadání nebo nápovědy mezi párovými značkami elementu text. Celý text mezi značkami se pak získá příkazem element.getTextContext(). Získaný text navíc podporuje HTML formátování, takže lze pro větší přehlednost v nápovědě využívat například dolní indexy pro označení veličin.

```
// nalezne vsechny elementy s nazvem edge
NodeList nList = doc.getElementsByTagName("edge");
for (int i = 0; i < nList.getLength(); i++) {
        org.w3c.dom.Node nNode = nList.item(i);
        if (nNode.getNodeType() == org.w3c.dom.Node.ELEMENT_NODE) {
                Element eElement = (Element) nNode;
                // zpracuje atribut id a vytvori instanci
                id = Integer.parseInt(eElement.getAttribute("id"));
                gEdge = new GraphEdge(id);
        }
}
```

Vzhledem k tomu, že některé vygenerované hodnoty jsou zobrazeny v zadání, viz kapitola 3.7, je vhodné, aby bylo zadání načteno až po vygenerování celého obvodu a pak se pouze na správná místa dosadily vygenerované hodnoty. Obdobně je to také při vygenerování nového zadání uživatelem.

3.11 Vytvoření teorie k elektrickým obvodům a spojení s rozšířenou realitou

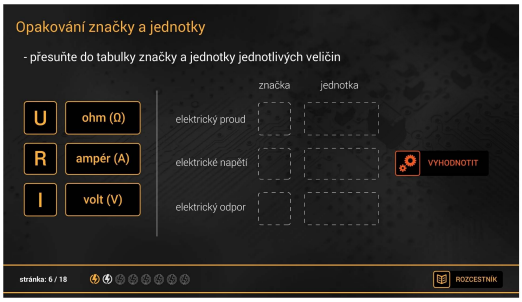
K vytvoření teorie jsem použil knihovnu android_core dodanou vedoucím práce. Slouží k vytvoření podobných výukových aplikací. Pro každou kapitolu je vytvořena aktivita (v mém případě Task01Activity, Task02Activity a Task03Activity), kde každou obrazovku teorie představuje jeden fragment. Mezi obrazovkami se pak uživatel přesouvá pomocí tzv. swipe (tažení prstu po obrazovce zleva doprava, případně zprava doleva). Definice jednotlivých obrazovek jsou uloženy v externím souboru mimo kód (v mém případě soubor questions.xml v adresáři res/raw). Jednoduché textové a informační obrazovky jsou celé zadefinované v externím souboru, složitější interaktivní úlohy se musí připravit zvlášť v kódu aplikace.

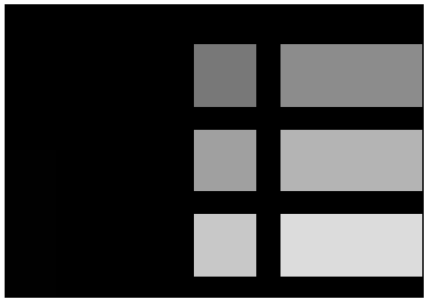
Jelikož úlohy rozšířené reality jsou aktivitou, bylo nutné připravit prostředí pro přechod mezi aktivitou kapitoly a aktivitou rozšířené reality (ARCircuitActivity). Vytvořil jsem tedy v teorii fragment pro každou úlohu rozšířené reality, v jehož parametrech byly uloženy všechny potřebné hodnoty (ID konfiguračního souboru, ID modelu, jméno image targetu) pro spuštění aktivity rozšířené reality. Uživateli se tedy po kliknutí na tlačítko „Spustit rozšířenou realitu“, viz obrázek 3.11, spustí nová aktivita s rozšířenou realitou podle uložených parametrů, která je zobrazena na popředí. Při přechodu zpět do teorie se vyvolá na popředí aktivita kapitoly.

Přetahovací úlohy popsané v kapitole 2.15 podporuje knihovna android_core. Celá úloha se vytvoří pomocí sady přetahovacích elementů (DraggableTextView), které může uživatel přetahovat po obrazovce. Na obrazovce je také umístěn obrázek, do kterého se mají jednotlivé elementy přetahovat. Pod tímto obrázkem je umístěna neviditelná šablona s různě barevnými plochami pro vyhodnocování. Každý element má pak definovanou barevnou plochu (barvu), do které se má přetáhnout. Vyhodnocení tedy probíhá tak, že se projdou všechny přetahovací elementy a zjistí se, jestli se jeho střed nachází na definované barvě. Pokud ano, je správně umístěn, jinak je umístěn špatně. Na obrázku 3.12 je zobrazen příklad obrázku, do kterého se mají elementy přetahovat (vlevo), a jeho šablona (vpravo).

Vzhledem k několika možným řešením složení vzorce Ohmova zákona do pyramidy, viz obrázek 3.13, jsem tuto úlohu rozšířil ještě o definice komutativních barev (třída Commu-

Obrázek 3.11: Obrazovka pro spuštění rozšířené reality

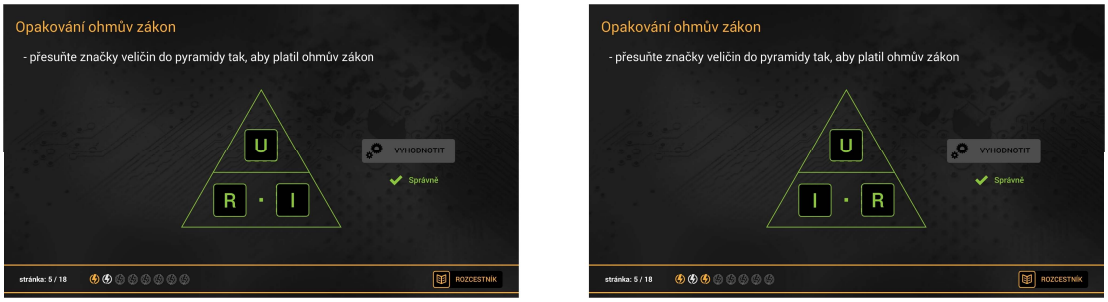




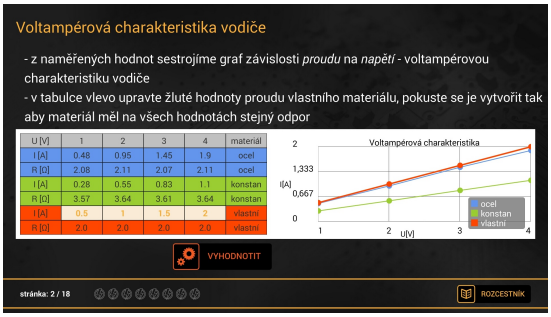
Obrázek 3.12: Příklad obrázku, do kterého se mají elementy přetahovat (vlevo), a jeho šablona (vpravo)

tativeDragDropQuestionFragment). Tedy pokud se některý element mohl nacházet na více místech (barvách), byly tyto barvy označeny jako komutativní a při vyhodnocení jsou správně označena i různá řešení. Poslední nutnou úpravou bylo držet si v další proměnné informaci o tom, jestli je barevná plocha již obsazená (v mém případě hash mapa, klíč barva, hodnota obsazeno/neobsazeno) tak, aby nebylo možné všechny komutativní elementy přesunout na jedno místo.

Pro interaktivní úlohu volt-ampérová charakteristika vodiče jsem využil knihovnu Android GraphView [10], která umožňuje vytváření grafů podle zadaných hodnot v prostředí Android. V tabulce vytvořené pomocí TableLayout jsem zadefinoval jeden řádek, v němž je možné měnit hodnoty proudu, které se poté automaticky vynáší do grafu. Díky tomu



Obrázek 3.13: Ohmův zákon - různá řešení přetahovací úlohy



Obrázek 3.14: Interaktivní úloha - VA charakteristika

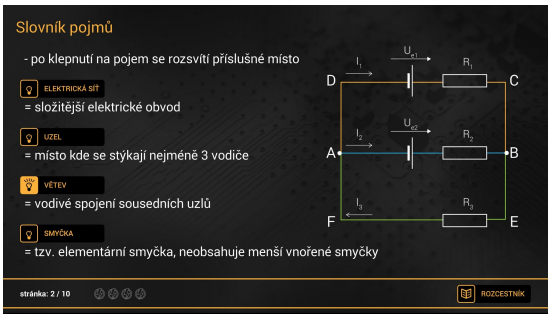
Obrázek 3.15: Interaktivní úloha - Zadávání rovnic, 2. Kirchhoffův zákon

může uživatel okamžitě vidět, jestli jsou doplňované hodnoty proudu lineární vůči danému napětí. Celý fragment je zobrazen na obrázku 3.14 a v mojí implementaci se jedná o třídu TableGraphFragment.

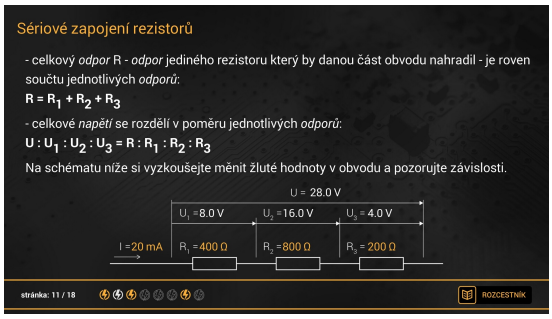
Další interaktivní úlohou bylo zadávání rovnic v Kirchhoffových zákonech (například pro uzel v síti). Vzhledem k velkému množství možných řešení jsem musel zvolit jiný přístup něž přetahovací úlohu. Vytvořil jsem tedy sadu tlačítek, pomocí kterých může uživatel zadat rovnici. Každé tlačítko je pak uvnitř vyhodnocování reprezentováno matematickou operací (+, -, ·), případně konkrétní hodnotou. Po zadání rovnice tak vznikne matematický výraz, který stačí vyhodnotit jako pravdivý (obě strany výrazu se rovnají) nebo nepravdivý (strany výrazu se nerovnají). K tomuto vyhodnocování je využita třída Expression z knihovny EvalEx [22]. Díky tomuto řešení může uživatel zadat libovolný výraz, a pokud je pravdivý je ohodnocen jako správný. Implementace tohoto fragmentu se nachází ve třídě ButtonsInputFragment a to, jak fragment v zařízení vypadá, je zobrazeno na obrázku 3.15.

Poslední dvě interaktivní úlohy slouží pouze k seznámení s pojmy (slovník pojmů u Kirchhoffových zákonů), případně k pochopení závislostí při zapojení rezistorů. Nejsou tedy nijak hodnocené. U slovníku pojmů se jedná pouze o jednoduchý interaktivní fragment, ve kterém se po kliknutí na tlačítko mění obrázek elektrické sítě na určitou dobu, viz obrázek 3.16, v uživateli je pak vzbuzen dojem, že se zvýrazní část elektrické sítě. V úloze pro pochopení závislostí zapojení rezistorů je možné měnit žluté hodnoty a ostatní jsou pak podle změny vypočítány a aktualizovány, viz obrázek 3.17. Toho je dosaženo pomocí rozhraní TextWatcher [17], které po každé změně žluté hodnoty zavolá metodu pro přepočítání všech ostatních hodnot. Fragmenty jsou definovány ve třídách InteractiveImageFragment a ResistorsInteractiveFragment.

Aby mohl mít pedagog při výuce přehled o tom, jak studenti plní jednotlivé úlohy, bylo vytvořeno jednoduché hodnocení v podobě blesků. Pokud se studentovi povede splnit úlohu (interaktivní úlohy a úlohy v rozšířené realitě) na první pokus, dostane za ní zlatý blesk. V případě, že splní úlohu na některý z dalších pokusů, dostane stříbrný blesk. Pokud úlohu student nesplní, zůstane příslušný blesk prázdný. Jednotlivé pokusy a stavy splněných úloh jsou ukládány ve třídě ApplicationState, konkrétně v hash mapách questionSlideTries a questionSlideStates. Příklad hodnocení je možné vidět na obrázku 3.17 nebo 3.11 (blesky vlevo dole).



Obrázek 3.16: Interaktivní úloha - slovník pojmů, Kirchhoffovy zákony



Obrázek 3.17: Interaktivní úloha - zapojení rezistorů

Kapitola 4

Testování

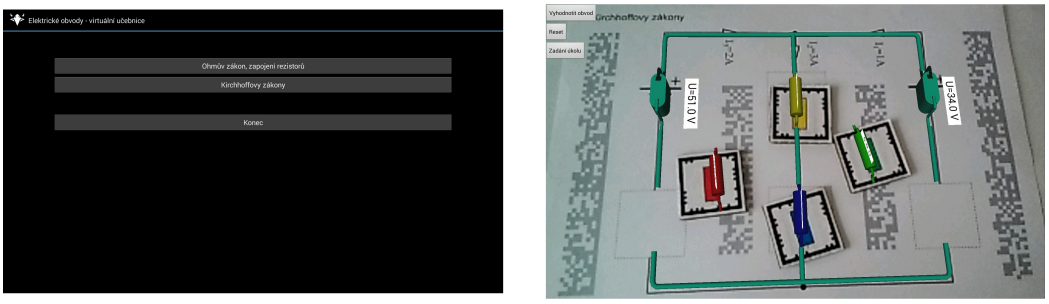
Testování aplikace probíhalo ve dvou fázích. V první fázi se testoval prototyp scénářů v rozšířené realitě a jeho hlavním účelem bylo zjistit, jestli jsou základní úkony při plnění scénáře pochopitelné pro uživatele. V druhé fázi se testovala téměř finální verze aplikace včetně hotové teorie a nasazené grafiky. Toto testování mělo sloužit spíše k identifikaci a odstranění drobných chyb v návrhu aplikace.

Cílová skupina pro testování byla daná ze zadání práce. Mělo se jednat o studenty středních škol, kteří mají v osnovách výuky fyziky zahrnuty elektrické obvody. Dále jsem u cílových uživatelů předpokládal základní počítačovou gramotnost a to, že nemají žádnou psychickou ani fyziologickou vadu (snížená hybnost, poruchy soustředění).

4.1 Testování prototypu

V tomto případě se jednalo o kvalitativní testování uživatelů, jehož cílem bylo ověření funkčnosti základních úkonů (úprava součástek, doplnění součástek do obvodu) plněných při práci se scénářem. Testování probíhalo v polovině května 2014 na střední škole v Kladně (Střední průmyslová škola stavební a Obchodní akademie) se studenty 3. ročníků, kteří se v průběhu školního roku (listopad, prosinec 2013) učili látku elektrických obvodů. Testování se zúčastnilo celkem čtrnáct studentů, z nichž šest plnilo úkoly samostatně (jednotlivci) a zbylých osm plnilo úkoly ve dvojicích. Testování dvojic jsme s vedoucím práce zvolili z toho důvodu, že jsme chtěli zjistit, jak by se studentům ve dvojicích pracovalo, jak si rozdělí práci a budou spolupracovat na řešení elektrického obvodu.

Testovací aplikace (prototyp) měla pouze základní rozhraní bez jakýchkoliv grafických elementů. Obsahovala obrazovku hlavního menu, ze kterého se přecházelo do dvou plněných scénářů (ve finální aplikaci nazvané „Ohmův zákon – pokročilá úloha“ a „Kirchhoffovy zákony – pokročilá úloha“), případně šlo ukončit aplikaci pomocí tlačítka „Konec“. Ve scénáři s rozšířenou realitou šlo pak zobrazit zadání úkolu, upravit součástku pomocí editoru součástek, doplnit součástky do obvodu a obvod vyhodnotit. Nebyla zde možnost zobrazit si nápovědu ke scénáři (tu částečně nahradila teorie na papírech, viz níže, případně moderátor testu), ani vygenerovat si vlastní zadání. Všichni studenti měli stejné zadání tak, aby se jim dalo snadněji napovídat v případě, že by si nevěděli rady s řešením úkolu. Byly zde také použity pouze základní testovací modely. Jak aplikace vypadala je zobrazeno na obrázku 4.1, vlevo



Obrázek 4.1: Vzhled prototypu aplikace, hlavní menu (vlevo), rozšířená realita (vpravo)

hlavní menu, vpravo obrazovka rozšířené reality se zobrazenými modely. Instalační soubor prototypu použitého při testování (ElectricalCircuits.apk) je uložen na přiloženém CD ve složce test\test_prototype\bin.

4.1.1 Struktura a průběh testování

Před vlastním testováním byli uživatelé seznámeni s účelem aplikace a celého testování. Poté jim byl předložen tzv. Dotazník před testováním, kde uživatelé vyplnili údaje o sobě (pohlaví a věk, školní prospěch), jestli používají mobilní zařízení a jaké mají zkušenosti s rozšířenou realitou. Konkrétní podoba dotazníku je v příloze C.1 společně s tabulkou vyplněných odpovědí. Jednotlivé vyplněné dotazníky jsou přepsány do elektronické podoby a uloženy na přiloženém CD ve složce test\test_prototype\pre_questionnaires. Po vyplnění dotazníku se přešlo k vlastnímu testování podle zadaných instrukcí, viz níže. Teorie, kterou měli uživatelé k dispozici, je v příloze C.2.

Instrukce k testování byly následující:

1. Před spuštěním aplikace se zběžně seznamte s teorií k látce přiložené na papírech (můžete jí využívat během celého testování).
2. Spusťte aplikaci „Elektrické obvody virtuální učebnice“, ikona aplikace je na hlavní obrazovce tabletu.
3. V hlavním menu vyberte úlohu „Ohmův zákon, zapojení rezistorů“, pozorně si přečtěte zadání úlohy a úlohu vyřešte.
4. Po úspěšném vyřešení úlohy se vraťte do hlavního menu.
5. V hlavním menu vyberte úlohu „Kirchhoffovy zákony“, pozorně si přečtěte zadání úlohy a úlohu vyřešte.
6. Po úspěšném vyřešení úlohy se vraťte do hlavního menu a ukončete aplikaci pomocí tlačítka „Konec“.

Po dokončení všech zadaných instrukcí uživatelé vyplnili tzv. Dotazník po testování, kde ohodnotili práci s aplikací, co se jim na aplikaci líbilo a co nelíbilo. Celý dotazník i se souhrnem odpovědí je v příloze C.4. Jednotlivé dotazníky jsou přepsány do elektronické podoby a uloženy na přiloženém CD ve složce test\test_prototype\post_questionnaires.

Všichni uživatelé uvedli, že používají chytrý telefon či tablet, devět z nich s OS Android, čtyři s iOS a pouze jeden s Windows Phone. Většina uživatelů neměla před testováním žádnou představu o rozšířené realitě, ale šest z nich po nápovědě znalo Google Glass či jinou aplikaci rozšířené reality.

Jako největší klady celé aplikace uživatelé uváděli hlavně použití moderních technologií a zobrazování 3D modelů na papíře, což jim přišlo zajímavé a atraktivní. Na aplikaci se uživatelům nelíbila hlavně práce s editorem a nečitelné hodnoty součástek, podrobně rozebírám tyto problémy v kapitole 4.1.2. Zajímavé také bylo, že všichni uživatelé (kromě jednoho) uvedli jako nejsložitější úkon během práce s aplikací výpočet úlohy, což byl také jeden z důvodů k vytvoření jednodušších scénářů, viz kapitola 2.3, a přidání nápovědy ke každému scénáři. Práci ve dvojici hodnotilo sedm uživatelů kladně, pouze jeden si stěžoval, že si chtěl více vyzkoušet práci s tabletem.

4.1.2 Popis nalezených problémů a návrh jejich řešení

Níže uvádím všechny důležité problémy, se kterými jsem se setkal během testování. Dále pak u každého problému popisuji návrh jeho řešení. Tyto nálezy vychází z vyplněných dotazníků a zápisů vzniklých při testování, viz příloha C.3.

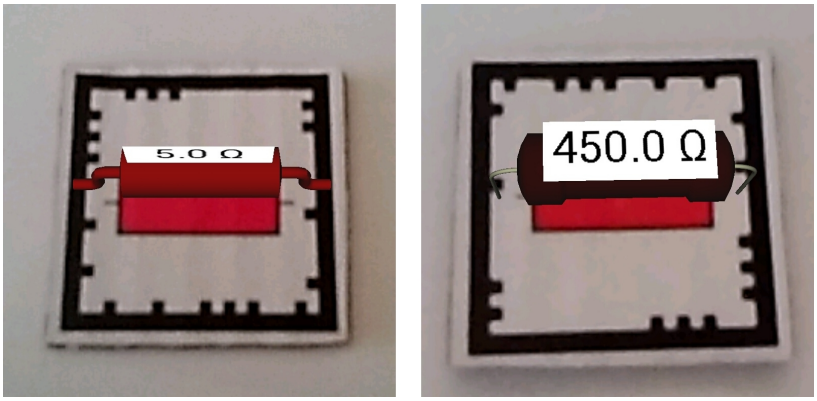
Zobrazující se dialog „Chcete editovat součástku“ po potvrzení úprav v editoru součástek

Jedním z velkých problémů při plnění úloh (kromě vlastního výpočtu úlohy) byly úpravy jednotlivých doplňovaných součástek. V několika případech (P2, P6, P9, P10) uživatelé vždy po upravení součástky a potvrzení úprav nechávali součástku v editoru součástek. Ta byla opět detekovaná jako v editoru součástek a ihned po potvrzení úprav se tedy zobrazil dialog „Chcete editovat součástku...“. Uživatelé až potom součástku vyjmuli z editoru a v dialogu stiskli tlačítko „Ne“. V jednom případě (P9) dokonce po zobrazení dialogu uživatel pouze vyměnil součástku za novou, kterou chtěl editovat a stiskl tlačítko „Ano“. Což způsobilo, že se upravovala součástka, která byla původně v editoru a ne nově vložená. Díky tomu uživatel poté špatně vyplnil obvod a byl zmatený z hodnot na jednotlivých součástkách.

Prvním řešením tohoto problému bylo vytvoření dialogu bezprostředně po potvrzení úprav součástky. Tento dialog vyzývá uživatele k vyjmutí součástky z editoru předtím než editor opustí. To ovšem výrazně zpomalilo práci s editorem díky dalšímu potvrzování dialogu. Proto jsem do dialogu přidal zaškrtávací možnost „Rozumím, tento dialog příště nezobrazovat“, po jejímž zaškrtnutí se zobrazování tohoto dialogu vypne. Dalším řešením problému bylo upravení dialogu před vstupem do editoru součástek. V tomto dialogu jsem nechal pouze jedno tlačítko „Upravit součástku“, které slouží pro vstup do editoru. Dialog také vyzývá k vyjmutí součástky z editoru v případě, že uživatel nechce součástku editovat. Po vyjmutí součástky z editoru dialog automaticky zmizí. Nemůže tak docházet k úpravám jiných součástek, než které jsou v editoru.

Špatně čitelné hodnoty jednotlivých součástek

Několika uživatelům (P1, P3, P4, P8) dělalo problém čtení hodnot na jednotlivých součástkách. To bylo způsobeno tím, že rovina s hodnotou součástky byla orientovaná na výšku



Obrázek 4.2: Porovnání čitelnosti rovin na rezistoru, prototyp (vlevo), finální verze (vpravo)

a uživatelé museli často naklánět tablet tak, aby si mohli přečíst aktuální hodnotu součástky. V některých případech dokonce vzali součástku do ruky, což způsobilo částečné zakrytí frame markeru a přerušení jeho detekce. Tím pádem zmizel 3D model součástky včetně roviny s hodnotou.

V dalším vývoji jsem tedy navrhl otočení roviny nad modelem tak, aby byla čitelná při pohledu svrchu na frame marker. To korespondovalo i s rovinami umístěnými na modelech jednotlivých scénářů a také výrazně zvýšilo čitelnost hodnoty součástky. Porovnání čitelnosti rovin na rezistoru v implementaci prototypu a finální verze aplikace je vidět na obrázku 4.2.

Označení prázdných míst v obvodu červeným kruhem

Jednoho uživatele (P1) také mátlo, že pokud je prázdné místo označeno červeným kruhem a vloží se do něj součástka, červený kruh nezmizí. V prototypu mizely červené kruhy až při opětovném kliknutí na tlačítko vyhodnotit. Uživatel si pak myslel, že do obvodu doplnil špatný typ součástky nebo součástku se špatnou hodnotou.

Z tohoto důvodu jsem navrhl změnu v algoritmu pro vyhodnocování vzájemné pozice frame markeru a image targetu. Pokud je některý z frame markerů vyhodnocen jako na pozici ve scénáři, zkontroluje se, jestli je viditelný červený kruh na pozici, a pokud ano, tak se zneviditelní.

Zůstávání přichycené součástky v obvodu, pokud byla odebrána bez detekce mimo obvod

Dalším funkčním problémem bylo zůstávání modelu přichycené součástky v obvodu, pokud se odebrala ze scénáře bez toho, aby byla detekována. Knihovna RajawaliVuforia totiž poskytuje pouze rozhraní, které posílá informaci o tom, že je frame marker detekován (metoda foundFrameMarker, viz kapitola 3.5). Neposkytuje ale žádné informace o tom, že některý z markerů přestal být detekován. Proto byl v prototypu aplikace vytvořen algoritmus, který uměl vyhodnotit umístění součástky mimo pozici pouze pokud byla mimo pozici detekována. Díky tomu vznikl funkční problém v situacích, kdy uživatel například doplnil součástku do obvodu na pozici a poté součástku zakryl a odebral zcela ze záběru kamery tabletu. Model

součástky v tomto případě zůstával přichycen k modelu scénáře, jelikož algoritmus nejprve správně vyhodnotil součástku jako na pozici, ale při jejím odebrání už jí nedetekoval.

Řešením tohoto problému byla úprava zobrazovacího cyklu modelu scénáře. Každá součástka dostala příznak o tom, jestli je nebo není detekovaná a ten se nastavil na true v případě, že byl příslušný frame marker detekován pomocí metody foundFrameMarker. Před vlastním zobrazením scénáře se pak projdou všechny součástky a v případě, že má součástka příznak nastavený na false, zneviditelní se její přichycený model. Po průchodu všech součástek se ještě nastaví příznak všech součástek na false tak, aby byly připraveny do dalšího zobrazovacího cyklu. Toto řešení mírně zhoršilo detekci (například ve špatných světelných podmínkách), jelikož nemusí být frame marker detekován v každém cyklu a dochází tak k přerušení přichycení součástky. To může způsobit potíže například při vyhodnocování obvodu (doplněná součástka není detekovaná a pozice je označena jako prázdná). V normálních podmínkách je ovšem detekce dostatečná a nedochází k žádným potížím.

Úloha Kirchhoffovy zákony v orientaci na výšku

Dalším nálezem byla chyba v návrhu scénáře „Kirchhoffovy zákony – pokročilá úloha“. Tento scénář měl všechny popisy veličin (napětí, proud) orientované na výšku. To nutilo uživatele k otáčení tabletu na výšku tak, aby se jim celý scénář pohodlně vešel do záběru kamery zařízení. Ovšem zbytek aplikace byl navržen pro orientaci na šířku (hlavní menu, tlačítka v rozšířené realitě, editor součástek). Kvůli tomu museli uživatelé během práce s aplikací otáčet tablet. Proto jsem scénář upravil tak, aby byl orientovaný na šířku. Také všechny ostatní, vzniklé po něm, jsou orientované na šířku.

Problémy se záměnami některých veličin

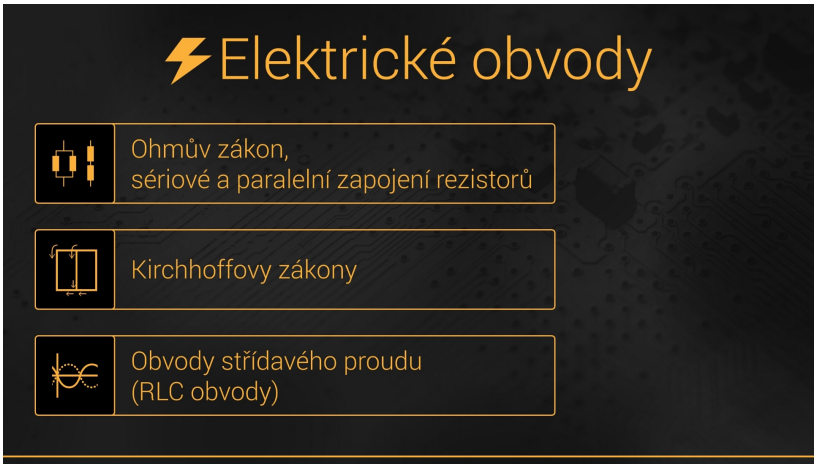
Někteří uživatelé (P1, P4) měli problém se záměnami veličin, například při přepisování hodnot napětí či proudu na papír. Při následném výpočtu a zpětném návratu k obvodu v rozšířené realitě nevěděli, které části obvodu veličiny náleží. Z toho důvodu jsem navrhl přidat ke stávajícímu označení indexy. Díky tomu je každá veličina jednoznačně určená a studenti by je neměli v obvodu zaměnit. Označení veličin také zjednodušilo popis jednotlivých nápověd, které byly přidány do aplikace v její další verzi tak, že se v ní studenti mohou snáze orientovat.

Nevýrazné tlačítko „Zadání“

Několika uživatelům (P3, P4, P6, P9) se při spuštění rozšířené reality obtížně hledalo tlačítko pro zobrazení zadání úkolu. Bylo to způsobeno jeho nevýraznou grafickou podobou. Tento problém byl vyřešen nasazením grafiky ve finální verzi aplikace s výraznou lištou na spodním okraji obrazovky zařízení.

4.2 Testování předfinální verze aplikace

Toto testování probíhalo formou tzv. field study, kdy byli uživatelé pozorováni při práci s aplikací přímo během jejího použití ve výuce. Testování probíhalo na začátku prosince 2014



Obrázek 4.3: Obrazovka hlavního menu - předfinální verze aplikace

ve dvou třídách na gymnáziu v Kralupech nad Vltavou (Dvořákovo gymnázium a Střední odborná škola ekonomická) se žáky 3. ročníků, kteří probírali látku Ohmova zákona a Kirchhoffových zákonů měsíc před vlastním testováním. V první třídě se zúčastnilo výuky patnáct studentů a její náplní byla kapitola „Ohmův zákon, sériové a paralelní zapojení rezistorů“, v druhé třídě se zúčastnilo výuky dvacet čtyři studentů a její náplní byla kapitola „Kirchhoffovy zákony“. V obou třídách prováděl výuku pan Kusák. Většina studentů pracovala ve dvojicích, pouze několik jedinců (převážně ti, kteří si chtěli aplikaci projít vlastním tempem bez ohledu na výuku) pracovalo samostatně.

Tato verze aplikace měla již nasazenou většinu grafiky, obsahovala hlavní menu, kde si uživatel mohl vybrat ze tří možných kapitol, zpracovanou teorii včetně interaktivních úloh a napojení na rozšířenou realitu. V rozšířené realitě byly zaneseny všechny návrhy zjištěné při předchozím testování, viz kapitola 4.1. Každý scénář v rozšířené realitě měl již kromě vlastního zadání i množinu nápověd, které mohl uživatel využít v případě, že si nebyl jistý postupem či některými používanými vzorci. Další přidanou možností scénáře oproti prototypu bylo vygenerovat si nové zadání. Jak vypadala obrazovka hlavního menu je na obrázku 4.3 a obrazovka scénáře v rozšířené realitě je například na obrázku 3.5. Instalační soubor aplikace použité při tomto testování je uložen na přiloženém CD ve složce test\test_prefinal\bin.

4.2.1 Struktura a průběh testování

Před vlastním testováním byli uživatelé seznámeni s účelem aplikace a s tím, že budou během výuky pozorováni. Poté jim byl předložen tzv. Dotazník před výukou, který byl obdobný jako při testování prototypu (věk, pohlaví, prospěch, zkušenosti z rozšířenou realitou). Konkrétní podoba dotazníku je v příloze D.1 stejně jako tabulka vyplněných odpovědí. Jednotlivé vyplněné dotazníky jsou naskenovány do elektronické podoby a uloženy na přiloženém CD ve složce test\test_prefinal\pre_questionnaires. Po vyplnění dotazníku se přešlo k vlastní výuce. Jak již bylo zmíněno výše, v každé třídě se vyučovala jiná kapitola aplikace. Pro účely seznámení s aplikací se ve druhé třídě, kde se vyučovaly Kirchhoffovy zákony, prošla kapitola Ohmův zákon do prvního scénáře v rozšířené realitě včetně. Tak, aby si studenti vyzkoušeli

jednoduchý scénář v rozšířené realitě, případně i práci s editorem součástek. V obou třídách si tedy studenti vyzkoušeli splnit minimálně dva scénáře v rozšířené realitě a několik interaktivních úloh.

Po ukončení výuky byl studentům předložen tzv. Dotazník po výuce. V tomto dotazníku studenti ohodnotili celkovou práci s aplikací, v rozšířené realitě pak práci s editorem součástek či nápovědou. Dále měli uvést, co se jim na aplikaci líbilo a co nelíbilo, zhodnotit interaktivní úlohy v teorii, případně jak se jim pracovalo ve dvojicích. Celý dotazník je zobrazen v příloze D.3 včetně souhrnu všech odpovědí. Jednotlivé dotazníky jsou naskenovány do elektronické podoby a uloženy na přiloženém CD ve složce test\test_prefinal\post_questionnaires.

Obdobně jako při testování prototypu téměř všichni uživatelé (kromě dvou) uvedli, že používají chytrý telefon nebo tablet. Většina z nich (cca 70 %) používá chytrý telefon s OS Android, cca 25 % využívá telefon s iOS a zbylých 5 % uvedlo, že využívá telefon s Windows Phone nebo OS Symbian. Pouze 10 % uživatelů mělo správnou představu o pojmu rozšířená realita a ti také uvedli správně i některou z aplikací, které využívají rozšířenou realitu.

Klady aplikace byly obdobné jako při testování prototypu. Uživatelé ocenili hlavně vzhled aplikace, 3D modely v rozšířené realitě, jednoduché ovládání, přesnost a přehlednost celé aplikace. Jediným problémem, který uživatelé uváděli opakovaně bylo dlouhé načítání modelů (10 % uživatelů). Tento problém podrobně rozebírám v kapitole 4.2.2. Oproti předchozímu testování se zlepšil názor na editor součástek. Pouze šest uživatelů uvedlo nějaký problém s editováním součástek, podrobně viz kapitola 4.2.2. Polovina uživatelů také použila během plnění scénářů v rozšířené realitě nápovědu a pouze třem uživatelům z celkových dvaceti, co nápovědu využili, nepřišla užitečná. Kladně byly také hodnoceny interaktivní úlohy, které připadaly uživatelům názorné a užitečné. Práce ve dvojici se většině uživatelů líbila, pouze cca 15 % si stěžovalo, že jim kolega tablet nepůjčil, případně s ničím nepomohl nebo neporadil.

4.2.2 Popis nalezených problémů a návrh jejich řešení

V této kapitole uvádím všechny podstatné nálezy zjištěné během výše zmíněného testování. Nálezy vycházejí z vyplněných dotazníků a z poznámek vytvořených během testování, viz příloha D.2. Každý nález pak zhodnocuji a případně zmiňuji návrh jeho řešení.

Interaktivní úloha – VA charakteristika, nevýrazný doplňovací řádek

Prvním problémem objeveným při výuce bylo doplnění hodnot proudu vlastního materiálu do tabulky VA charakteristiky. Problém byl zapříčiněn nevýrazným řádkem, do kterého se měly doplnit hodnoty. Uživatelé pak zmateně klikali všude po tabulce ve snaze někam umístit kurzor a dopsat hodnotu. Navrhl jsem tedy zesvětlení řádku, do kterého se mají doplnit hodnoty, tak aby se odlišil od ostatních hodnot v tabulce. Dále jsem text hodnot tučně zvýraznil a obarvil žlutou barvou. V popisu úlohy jsem pak přidal text, že se mají změnit žluté hodnoty v tabulce. Díky tomu je doplňovací řádek odlišený a uživatel na první pohled rozezná, o jaký řádek se jedná.

Interaktivní úloha – VA charakteristika, nevyjíždějící tabulka při focusu na upravovanou hodnotu

V případě, že uživatel chtěl upravit danou hodnotu v tabulce, klikl na ní a zároveň s vysunutím softwarové klávesnice se posunul celý layout včetně tabulky tak, aby daná hodnota byla nad klávesnicí (toho je dosaženo díky ScrollView layoutu, který se posouvá při vysunutí klávesnice). Problém ovšem nastal v případě, kdy uživatel zasunul klávesnici (například pomocí tlačítka zpět) a opětovným kliknutím na stejnou hodnotu chtěl znovu vysunout klávesnici. V tomto případě, kdy se nezměnil focus na jiný element (EditText s upravovanou hodnotou), se vysunula pouze klávesnice a layout zůstal na místě. To zapříčinilo překrytí části layoutu klávesnicí a uživatel tak neviděl hodnotu, kterou upravoval.

Jedná se nejspíše o vnitřní chybu Android SDK, a proto jsem musel navrhnout řešení, které by obešlo tento problém. Vytvořil jsem si tedy vlastní EditText (zděděním od třídy EditText, v mém projektu nazvaná MyEditText), kterému se jako jeden z parametrů předává jiný element (v mém případě například tlačítko „Vyhodnotit“ pod tabulkou). Dále jsem ve třídě přetížil metodu onKeyPreIme, která odchytává událost skrytí softwarové klávesnice. V případě, že je odchycena tato událost, tak se změní focus v layoutu na tlačítko „Vyhodnotit“. Díky tomu se při opětovném vysunutí klávesnice posune i celý layout včetně tabulky.

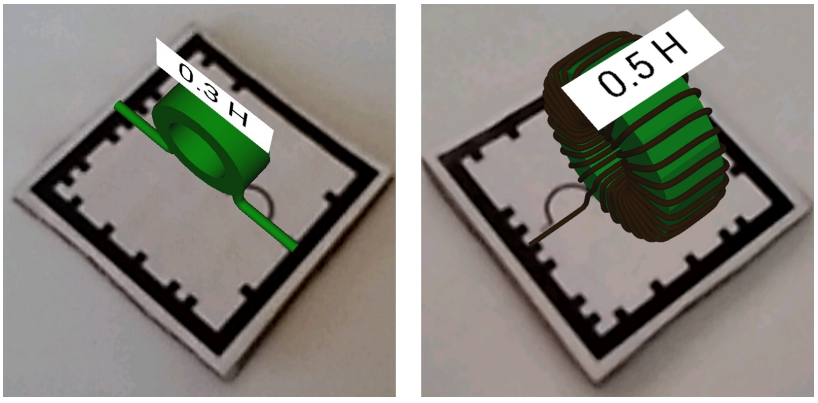
Interaktivní úloha – Zapojení rezistorů, při smazání celé hodnoty nelze již upravit

Další problém byl objeven v interaktivních úlohách se zapojením rezistorů. Pokud uživatel smazal celou hodnotu, kterou mohl upravovat (například hodnotu odporu) tak, že její délka byla nulová, a opustil dané pole pro zadávání například kliknutím do jiného pole pro zadávání, případně skrytím softwarové klávesnice, už nebylo možné toto pole upravit. Tento problém byl způsoben tím, že vstupní pole (EditText) má dynamickou šířku podle toho, jak dlouhá je v něm hodnota. Proto, když uživatel smazal celou hodnotu (hodnota měla nulovou délku), i dané vstupní pole mělo nulovou šířku a nešlo do něj kliknout.

Navrhl jsem tedy obdobné řešení jako je popsáno v předchozím problému. Vytvořil jsem si opět vlastní EditText (v implementaci třída ResistorEditText), kterému jsem opět přetížil metodu onKeyPreIme. V tomto případě se ale při odchycení události zjišťuje délka hodnoty vstupního pole a, pokud je nulová, upraví se na řetězec „0.0“. Obdobná úprava se provede i v případě, kdy byl změněn focus vstupního pole na jiné, pomocí rozhraní OnFocusChangeListener (metoda onFocusChange). Díky tomu není možné, aby mělo vstupní pole nulovou délku a uživatel do něj tedy může kdykoliv kliknout a hodnotu upravit.

Rozšířená realita – Dlouhé načítání modelů

Několik uživatelů uvedlo jako jednu z věcí, která se jim na aplikaci nelíbila, dlouhé načítání modelů. Díky nové verzi modelů, které jsou složitější než byly modely v prototypu aplikace, se doba načítání modelů zdvojnásobila na cca 30 vteřin. Největší vliv na dobu načítání má model induktoru, který má v porovnání s modelem použitým v prototypu složité vinutí, viz obrázek 4.4. Velikost souboru tohoto modelu je řádově (10×) větší než soubory s modely rezistoru či kondenzátoru a přibližně 2× větší než většina modelů celých scénářů. Jeho zjednodušením, například zobrazením vinutí jako textury na těle induktoru, by se zkrátila doba načítání modelů cca na polovinu.



Obrázek 4.4: Porovnání složitosti modelu induktoru, prototyp (vlevo), finální verze (vpravo)

Vzhledem k tomu, že si na tento problém stěžovalo relativně malé procento (10 %) uživatelů, rozhodl jsem se ponechat modely ve stávající složitosti. Pokud by se na tento problém ve výuce naráželo častěji, jeho vyřešení není nijak složité (vytvoření nového modelu induktoru, a záměna dvou souborů v projektu aplikace).

Rozšířená realita – Kirchhoffovy zákony, špatná pozice textu se součtem rezistorů

Během plnění scénáře „Kirchhoffovy zákony – základní úloha“ v rozšířené realitě se objevil problém s dosazením hodnoty součtu rezistorů do zadání. Hodnota byla dosazována na špatnou pozici, což bylo způsobeno pravděpodobně některou s předchozích úprav zadání scénáře (například změna názvu úlohy). Řešením bylo opravení konfiguračního souboru tak, aby byla hodnota dosazena na správnou pozici.

Rozšířená realita – Špatná detekce editoru součástek

Tři dvojice uživatelů uvedly v dotazníku, že se jim špatně pracovalo s editorem součástek z důvodu jeho špatné detekce, neostření tabletu či nevěděly, jak ho použít. Problém špatné detekce a neostření byl zapříčiněn špatnými světelnými podmínkami v konkrétním místě, kde dvojice pracovala (například přímo pod zářivkou, kde si uživatel stíní vlastním tabletem). Tento problém není možné odstranit jinak než zlepšením světelných podmínek, případně upravením pracovní polohy tak, aby si uživatel nestínil. Druhý problém (použití editoru) lze vyřešit například videem z práce s aplikací před prvním scénářem v rozšířené realitě. V tomto videu by bylo ukázáno, jak pracovat se scénářem, jak upravit hodnoty součástek pomocí editoru součástek a následně je doplnit do obvodu.

Kapitola 5

Závěr

Cílem této práce bylo analyzovat existující řešení pro rozšířenou realitu na OS Android. Vzhledem ke zvoleným kritériím a s přihlédnutím k mým zkušenostem z předchozího studia jsem zvolil knihovnu Vuforia společně s rozšířeními Rajawali a RajawaliVuforia. Navrhl jsem koncept v rozšířené realitě pro doplňování součástek do elektrického obvodu a jeho vyhodnocování, který jsem během řešení práce rozšířil o možnost úpravy součástek a generování zadání do obvodu. Po konzultacích s vyučujícími na středních školách jsem tento koncept aplikoval na osm scénářů, které pokrývají většinu látky elektrických obvodů probíranou na středních školách. Scénáře jsem rozdělil do třech kapitol (Ohmův zákon, Kirchhoffovy zákony, Obvody střídavého proudu) a ke každé kapitole přidal teoretický základ nutný k pochopení dané látky.

Výsledkem této práce je tedy mobilní aplikace pro tablety s OS Android, která využívá navržený koncept. V práci jsem musel zvolit vhodnou reprezentaci elektrického obvodu v paměti zařízení, analyzovat jaké objekty použít pro detekci součástek a scénářů a implementovat výpočet jejich vzájemné polohy tak, aby bylo možné celý obvod pomocí aplikace automaticky vyhodnotit. Algoritmus pro výpočet vzájemné polohy jsem také použil pro úpravy jednotlivých součástek. Dále jsem pro každý scénář vytvořil konfigurační soubor ve formátu XML, ze kterého se načítají všechny parametry daného scénáře. Také jsem vytvořil sadu modelů, kterou jsem vyexportoval do formátu OBJ, a pomocí rozšíření Rajawali jsem implementoval jejich načítání do scény v rozšířené realitě. Nakonec jsem vytvořil teorii ke všem kapitolám a přidal i několik interaktivních úloh, které měly studenty zkoušet ze základních teoretických znalostí.

Testování s uživateli probíhalo ve dvou fázích. V první fázi se testoval pouze prototyp aplikace se dvěma scénáři v rozšířené realitě a účelem tohoto testování bylo ověření funkčnosti základních úkonů plněných během práce se scénářem. Ukázalo se, že největší problém mají uživatelé s úpravou součástek, proto jsem tento úkon v dalším vývoji aplikace mírně změnil tak, aby byl co nejjednodušší. Ve druhé fázi se testovala již téměř finální verze aplikace. Během tohoto testování byly nalezeny spíše drobné chyby hlavně v teorii a interaktivních úlohách. V tomto testování jsem také ověřil, že změněný způsob úprav součástek je již pro uživatele jasný a jednoduchý. Testování také ukázalo, že většina uživatelů nemá žádné předchozí zkušenosti s rozšířenou realitou a ani neví, co si pod pojmem rozšířená realita představit. Pro většinu testovaných uživatelů byla tedy práce s mojí aplikací první zkušeností s rozšířenou realitou. Na aplikaci uživatelé oceňovali hlavně její vzhled, třetí rozměr v podobě 3D modelů

v rozšířené realitě, použití moderních technologií a jednoduché ovládání. Někteří dokonce uvedli, že díky aplikaci konečně pochopili probíranou látku.

Aplikaci by šlo rozšířit o několik dalších scénářů (například jednoduché scénáře v obvodech střídavého proudu s jednou součástkou), případně přidat i další kapitolu o polovodičích. Ta by vyžadovala kromě výpočtu vzájemné polohy součástky a scénáře i výpočet vzájemné orientace (například při umístění diody v propustném/nepropustném směru). Tento výpočet jsem ve své práci také implementoval, bohužel se mi ho ale nepovedlo odladit tak, aby bylo přichytávání součástek ke scénáři spolehlivé. Nicméně ve stávajících scénářích není orientace součástky podstatná, tudíž nebyl výpočet vzájemné orientace potřebný. Při konzultaci s jedním z pedagogů také vznikla myšlenka vytvořit testovací kapitolu pouze se scénáři v rozšířené realitě. Uživatel by si mohl před vstupem do kapitoly vybrat, které scénáře chce procvičovat, a k jednotlivým scénářům by pak byla připojena statistika o tom, kolikrát ho uživatel splnil správně a kolikrát špatně. Dalším rozšířením by mohlo být posílání statistik o splněných scénářích či interaktivních úlohách pedagogovi na email. Díky tomu by měl rychlou zpětnou vazbu o tom, zda studenti danou látku chápou, případně jestli se postupně zlepšují, bez nutnosti je jednotlivě obcházet ve třídě.

Myslím, že výsledná aplikace je vhodná jako doplněk výuky elektrických obvodů v hodinách fyziky na středních školách. Mírným úskalím může být to, že použití aplikace vyžaduje relativně dobré světelné podmínky a nutnost přípravy všech papírových materiálů. Testování ovšem ukázalo, že je aplikace za dobrých světelných podmínek velice přesná a spolehlivá a že se jedná o atraktivní a zábavnou formu výuky. Jen samotná práce na tabletech ve výuce studenty baví a přidání nového pro většinu studentů neznámého elementu, jako je rozšířená realita výuku ještě více zatraktivní. Dalším kladným elementem zjištěným při testování byla práce ve dvojicích. Pokud si studenti práci rozdělí, případně se v plnění jednotlivých úkonů vystřídají, oceňují na práci ve dvojici vzájemnou spolupráci a to, že se mohou navzájem doplňovat.

Seznam obrázků

Seznam tabulek

Literatura

[1] ARToolWorks. ARToolKit for Android. <http://www.artoolworks.com/products/artoolkit-for-mobile/artoolkit-for-android/>, 2014. [online], [cit. 2015-1-4].

[2] Autodesk, Inc. Autodesk Maya. <http://www.autodesk.com/products/autodesk-maya/overview>, 1998-2014. [online], [cit. 2015-1-4].

[3] Beyond Reality. IN2AR. <http://in2ar.com/>, 2014. [online], [cit. 2015-1-4].

[4] Blender Foundation. Blender. <http://www.blender.org/>, 1995-2014. [online], [cit. 2015-1-4].

[5] BRAY, T. et al. Extensible markup language (XML) 1.0 (Fifth Edition). *W3C Recommendation 26 November 2008. http://www.w3.org/TR/xml/*. 2008.

[6] CORMEN, T. H. et al. *Introduction to algorithms*. 2. Cambridge, MA : MIT press Cambridge, 2001.

[7] Cygnus Solutions. Cygwin. <http://cygwin.com/index.html>, 1995-2014. [online], [cit. 2015-1-4].

[8] DAQRI LLC. Anatomy 4D. <http://daqri.com/project/anatomy-4d/>, 2014. [online], [cit. 2015-1-4].

[9] Eclipse Foundation. Eclipse IDE. <http://www.eclipse.org/>, 2001-2014. [online], [cit. 2015-1-4].

[10] GEHRING, J. Android GraphView. <http://android-graphview.org>, 2014. [online], [cit. 2015-1-4].

[11] Google Inc. Android Development Tools. <http://developer.android.com/tools/sdk/eclipse-adt.html>, 2009-2014. [online], [cit. 2015-1-4].

[12] Google Inc. AsyncTask - Developer documentation. <http://developer.android.com/reference/android/os/AsyncTask.html>, 2014. [online], [cit. 2015-1-4].

[13] Google Inc. Canvas - Developer documentation. <http://developer.android.com/reference/android/graphics/Canvas.html>, 2014. [online], [cit. 2015-1-4].

[14] Google Inc. Android NDK. <http://developer.android.com/tools/sdk/ndk/index.html>, 2009-2014. [online], [cit. 2015-1-4].

[15] Google Inc. Parcelable - Developer documentation. <http://developer.android.com/reference/android/os/Parcelable.html>, 2014. [online], [cit. 2015-1-4].

[16] Google Inc. StaticLayout - Developer documentation. <http://developer.android.com/reference/android/text/StaticLayout.html>, 2014. [online], [cit. 2015-1-4].

[17] Google Inc. TextWatcher - Developer documentation. <http://developer.android.com/reference/android/text/TextWatcher.html>, 2014. [online], [cit. 2015-1-4].

[18] HIRZER, M. Marker detection for augmented reality applications. *Inst. For Computer Graphics and Vision, Graz University of Technology, Austria.* 2008.

[19] IPPEL, D. Rajawali. <https://github.com/masdennis/rajawali>, 2012-2014. [online], [cit. 2015-1-4].

[20] IPPEL, D. RajawaliVuforia. <https://github.com/MasDennis/RajawaliVuforia>, 2013-2014. [online], [cit. 2015-1-4].

[21] KATO, H. – BILLINGHURST, M. Marker tracking and hmd calibration for a video-based augmented reality conferencing system. *2nd IEEE and ACM International Workshop on.* 1999, s. 85–94.

[22] KLIMASCHEWSKI, U. EvalEx - Java Expression Evaluator. <https://github.com/uklimaschewski/EvalEx>, 2014. [online], [cit. 2015-1-4].

[23] LEPIL, O. *Fyzika pro gymnázia.* Praha : Galaxie, 2. vyd. edition, 1993.

[24] Metaio GmbH. Metaio. <http://www.metaio.com/>, 2003-2014. [online], [cit. 2015-1-4].

[25] MURRAY, J. D. – VANRYPER, W. *Encyclopedia of graphics file formats.* Second Edition. Sebastopol, CA : O'Reilly Media, 1996.

[26] OGNYANOV, N. NioGraphs - A Library of Graph Algorithms. <https://code.google.com/p/niographs/>, 2013. [online], [cit. 2015-1-4].

[27] Oracle. DOM Parser - API Specification. <https://docs.oracle.com/javase/7/docs/api/javax/xml/parsers/DocumentBuilder.html>, 2014. [online], [cit. 2015-1-4].

[28] Qualcomm Connected Experiences, Inc. Frame Marker. <https://developer.vuforia.com/resources/dev-guide/frame-markers>, 2011-2014. [online], [cit. 2015-1-4].

[29] Qualcomm Connected Experiences, Inc. Image Targert. <https://developer.vuforia.com/resources/dev-guide/image-targets>, 2011-2014. [online], [cit. 2015-1-4].

[30] Qualcomm Connected Experiences, Inc. Vuforia. <https://developer.vuforia.com/>, 2011-2014. [online], [cit. 2015-1-4].

[31] ROLLAND, J. P. – HOLLOWAY, R. L. – FUCHS, H. Comparison of optical and video see-through, head-mounted displays. In *Photonics for Industrial Applications*, s. 293–307. International Society for Optics and Photonics, 1995.

[32] StatCounter. Top 8 Mobile & Tablet Operating Systems. <http://gs.statcounter.com/#mobile+tablet-os-ww-monthly-201311-201411>, 2014. [online], [cit. 2015-1-4].

[33] TARJAN, R. Enumeration of the elementary circuits of a directed graph. *SIAM Journal on Computing*. 1973, 2, 3, s. 211–216.

[34] The Inkscape Team. Inkscape. <https://inkscape.org/cs/>, 2003-2014. [online], [cit. 2015-1-4].

[35] KLEIN, R. – BOONSTRA, C. – LENS-FITZGERALD, M. Layar. <https://www.layar.com/>, 2009-2013. [online], [cit. 2015-1-4].

[36] Wikitude GmbH. Wikitude. <http://www.wikitude.com/>, 2009-2013. [online], [cit. 2015-1-4].

[37] Wikitude GmbH. Wikitude Places - Sony Select. <https://play.google.com/store/apps/details?id=com.wikitude.places>, 2013. [online], [cit. 2015-1-4].

Příloha A

Seznam použitých zkratek

3D Troj dimenzionální (Three Dimensional)

ADT Android Development Tools

API Application Programming Interface

AR Rozšířená realita (Augmented Reality)

CSS Kaskádové styly (Cascading Style Sheets)

FBX Filmbox

GPS Globální polohovací systém (Global Positioning System)

HTML HyperText Markup Language

HW Hardware

ID Identifikační číslo

IDE Integrated Development Environment

JPEG Joint Photographic Experts Group

MB Megabyte

MTL Material Template Library

NDK Native Development Kit

OBJ Wavefront OBJ

OS Operační systém (Operating System)

RGB Red, Green, Blue

RLC Rezistor, induktor, kondenzátor

SDK Software Development Kit

XML Extensible Markup Language

Příloha B

Obsah přiloženého CD

Níže je stručně popsán obsah a adresářová struktura přiloženého CD.

```
├── bin ..................... instalační soubor aplikace, pdf se všemi papírovými podklady
├── doc ............................... vygenerovaná dokumentace projektu v JavaDoc
│   └── index.html ............................................ index dokumentace
├── media
│   ├── models
│   │   ├── blender ........ finální verze modelů vytvořená v Blenderu se všemi podklady
│   │   └── maya ............................... testovací verze modelů vytvořená v Maye
│   └── targets
│       ├── frame_markers ........... vektorové podklady a bitmapy ke frame markerům
│       └── image_targets ............ vektorové podklady a bitmapy k image targetům
├── src
│   ├── libs ............................ knihovny pro přiložení do pracovního prostoru
│   └── project ... projekt ElectricalCircuits v Eclipse IDE se všemi zdrojovými soubory
├── test
│   ├── test_prefinal
│   │   ├── bin ......................... instalační soubor aplikace použité při testování
│   │   ├── media ............................ papírové podklady použité při testování
│   │   ├── photos ....................... několik fotografií pořízených během testování
│   │   ├── post_questionnaires ................... naskenované dotazníky po testování
│   │   └── pre_questionnaires ............... naskenované dotazníky před testováním
│   └── test_prototype
│       ├── bin ......................... instalační soubor aplikace použité při testování
│       ├── media ............................ papírové podklady použité při testování
│       ├── photos ....................... několik fotografií pořízených během testování
│       ├── post_questionnaires ..................... přepsané dotazníky po testování
│       └── pre_questionnaires ................... přepsané dotazníky před testováním
├── text
│   ├── tex_src ....................................... zdrojové soubory tohoto textu
│   └── rozsirena_realita_pro_vyuku.pdf ................... tento text ve formátu pdf
└── README.txt ..................................... popis struktury přiloženého CD
```

Příloha C

Testování prototypu

V této kapitole jsou zobrazeny dotazníky před a po testování prototypu i se shrnutím jejich výsledků. Dále pak teorie, kterou měli uživatelé k dispozici během celého testování (měla částečně nahradit stávající teorii a nápovědu ve finální verzi aplikace), a také přepis poznámek vytvořených během testování moderátorem testu.

C.1 Dotazník před testováním a jeho výsledky

Dotazník před testováním – Elektrické obvody, virtuální učebnice

1. Vaše pohlaví: muž x žena

2. Váš věk:

3. Uveďte prosím Vaší poslední známku na vysvědčení z fyziky a matematiky:

   Fyzika:

   Matematika:

4. Jaké používáte mobilní zařízení? (mobilní telefon, tablet, apod.)

5. Co si představíte pod pojmem rozšířená realita?

6. Jaké konkrétní aplikace rozšířené reality znáte?

Tabulka C.1: Výsledy dotazníku před testováním, (P1 až P6 jsou participanti pracující jako jednotlivci, ostatní pracovali ve dvojicích)

	Pohlaví	**Věk**	**Prospěch fyzika**	**Prospěch matematika**	**Používaná mobilní zařízení**	**Pojem rozšířená realita**	**Aplikace rozšířené reality**
P1	žena	17	2	2	Nokia smartphone	něco víc okolo reality	x
P2	žena	17	1	1	Android smartphone	x	x
P3	muž	19	3	4	Android smartphone, tablet Android + iOS	x	x
P4	žena	18	2	3	Android startphone, Android tablet	x	x
P5	muž	19	4	4	Android smartphone	virtuální svět	výcvik vojáků
P6	muž	20	3	4	Android smartphone	něco s virtuální realitou?	Google Glass
P7a	muž	18	3	4	Android smartphone	x	x
P7b	muž	18	2	2	Android smartphone, Android tablet	x	Google Glass
P8a	žena	17	3	3	iOS smartphone	x	x
P8b	muž	18	2	2	iOS smartphone	realita + imaginární věci	vizualizace
P9a	muž	17	4	3	iOS smartphone	x	x
P9b	žena	17	2	2	Android smartphone	x	x
P10a	muž	18	3	3	Android smartphone	x	Google Glass
P10b	muž	18	4	4	iOS smartphone, Android tablet	x	Google Glass

C.2 Teorie pro uživatele

Složitější elektrické obvody vytvářejí elektrické sítě. Důležité součásti sítí jsou uzly a větve. Uzel elektrické sítě je místo, kde se vodivě stýkají nejméně tři vodiče. Soustava prvků elektrického obvodu zapojených mezi dvěma sousedními uzly (vodivé spojení sousedních uzlů) se nazývá větev. Když větve na sebe navazující vytvářejí uzavřený obvod, mluvíme o smyčce.

Při výpočtu neznámých proudů ve větvích a napětí na jednotlivých rezistorech, známe–li napětí zdrojů a odpory rezistorů, nebo při určování neznámých odporů rezistorů při známých hodnotách napětí a proudů používáme Kirchhoffovy zákony.

Jsou jedním ze základních nástrojů při teoretické analýze obvodů. Zákony byly poprvé popsány roku 1845.

1. Kirchhoffův zákon (pro uzel elektrické sítě):

Součet všech proudů vstupujících do uzlu nebo součástky je roven součtu všech proudů vystupujících z uzlu nebo součástky (tj. proud se nikde nehromadí)

2. Kirchhoffův zákon

Orientovaný součet všech napětí ve smyčce je nulový.

Ohmův zákon: napětí na odporu je součinem jeho hodnoty a proudu, který skrze něj protéká.

Analýza uzlů: v obvodu se najdou a označí všechny uzly. Libovolně zvolenému uzlu se přiřadí nulový elektrický potenciál. Všem zbývajícím se přiřadí neznámá napětí oproti referenčnímu uzlu. Pro každý z uzlů kromě referenčního se sestaví rovnice podle 1. Kirchhoffova zákona. Tato soustava rovnic se poté vyřeší.

Analýza smyček: Na schématu se najdou elementární smyčky, tzn. smyčky, které neobsahují menší vnořené smyčky. Každé takové smyčce se přidělí proud, který jí prochází. Pro každou smyčku se zapíše rovnice podle 2. Kirchhoffova zákona, ve které se jako neznámá použije proud protékající smyčkou. Tato soustava rovnic se poté vyřeší.

C.3 Přepis poznámek vytvořených při testování

Níže jsou uvedeny stručné přepisy poznámek pořízených během testování. Tučně jsou zvýrazněny problémy objevené během testování.

Participant 1

Úloha Ohmův zákon:

- řeší obvod po částech, opíše proud dopočítá odpor, prohlíží si součástky v rozšířené realitě, hledá ten co má správný odpor, poté zkouší editor součástek
- **dívá se na součástky shora, popisky rezistorů nejsou vidět, naklání součástku rukou**
- vkládá rezistor do Editoru součástek, poté míří kamerou na editor.
- drží tablet vysoko a pod úhlem

- **v obvodu zůstává cívka, kterou tam předtím omylem vložila**
- **po vložení součástky na červené místo je místo stále červené**

Úloha Kirchhoffovy zákony:

- **používá tablet na výšku (úloha na výšku), ovšem zbytek aplikace je na šířku**
- **nevypisuje si proměnné, pouze čísla na papír, poté zamění veličiny**
- většina času výpočet úlohy
- ke konci úlohy nervózní, nejdou výpočty
- $R_{23} = 11$, ale maximální možná hodnota je 10, neví si rady

Participant 2

Úloha Ohmův zákon:

- pochopila jak použít tablet a zobrazila scénář
- v jedné ruce drží tablet druhou si přepsala hodnoty ze scénáře
- nepochopila jak použít editor součástek, po nápovědě použití bez problému
- **nechala součástku v editoru, po návratu z editoru opět vyskočil dialog, dala ne**
- po menší nápovědě dopočítala R_4, zbylé spočítala bez nápověd

Úloha Kirchhoffovy zákony:

- přepis hodnot, drží tablet jednou rukou uprostřed, druhou přepisuje hodnoty na papír
- velká nápověda s řešením, pomoc se sestavením první rovnice, druhou sestavila již sama
- špatně nastavená jedna součástka, všimla si kde je chyba a součástku opravila

Participant 3

Úloha Ohmův zákon:

- **neví kde je zadání úkolu, a jak poznat hodnotu na rezistoru**
- neví jak použít editor součástek, po nápovědě se zalíbilo
- jeden rezistor špatně nastavený, chyby si všiml a doplnil správně
- **obtížně se mu čtou hodnoty na rezistoru, naklání tablet, poté i součástku**

Úloha Kirchhoffovy zákony:

- nutná nápověda, osvěžení 2. Kirchhoffova zákona, první smyčka sestavena s pomocí, druhou sestavil sám
- problémy s výpočtem úlohy
- správně rozdělil R_{23}, dokonce použil jednu hodnotu bez editace (5) a druhou nastavil na 6

Participant 4

Úloha Ohmův zákon:

- opisuje s lokty na stole, drží tablet a opisuje hodnoty
- nepoužívá papír s teorií
- po vypočítání úlohy nastavuje a vkládá rezistory
- **problém se záměnami různých hodnot napětí, proudů**

Úloha Kirchhoffovy zákony:

- **nemůže najít zadání vrací se do menu**
- **naklání tablet, aby viděl i hodnoty na rezistorech**

Participant 5

Úloha Ohmův zákon:

- neví jak začít, po chvíli začne opisovat hodnoty na papír
- doplnění do obvodu a jeho vyhodnocení v pořádku

Úloha Kirchhoffovy zákony:

- OK

Participant 6

Úloha Ohmův zákon:

- v menu vybral špatný úkol, vrací se zpět do menu, **nemůže najít zadání**
- nepochopil jak použít editor, po nápovědě v pořádku
- držení tabletu oběma rukama, dívá se z úhlu kvůli hodnotám na rezistorech
- **neustále nechává součástku v editoru, a vyskakuje na něj dialog**
- jedna součástka špatně vypočítaná, po opravě vše v pořádku

Úloha Kirchhoffovy zákony:

- práce s tabletem a editorem OK
- problémy se sestavením rovnic, po sestavení 1. s nápovědou skládá 2. rovnici sám

Participanti 7a,b (dvojice)

Úloha Ohmův zákon:

- a drží tablet, b opisuje hodnoty na papír
- přišli na to jak funguje editor součástek
- výpočet hodnot rezistorů zpaměti, vše správně
- doplnění a vyřešení obvodu bez problému

Úloha Kirchhoffovy zákony:

- problémy se sestavením rovnic, po sestavení 1. s nápovědou skládá 2. rovnici sám

Participanti 8a,b (dvojice)

Úloha Ohmův zákon:

- a drží tablet, b zkouší vložit rezistor
- a drží a diktuje hodnoty, b přepisuje na papír
- b míří na editor, a edituje a potvrzuje
- **a musí natáčet tablet aby byly vidět popisky**

Úloha Kirchhoffovy zákony:

- a edituje, b dává do editoru

Participanti 9a,b (dvojice)

Úloha Ohmův zákon:

- **hledají tlačítko zadání úkolu**
- b drží a opisuje
- b edituje součástky v aplikaci, a dává a vyndavá součástky z editoru
- špatně si opsali hodnoty

- **vyskakuje dialog po nechaném rezistoru v editoru součástek, pouze vymění součástku a dají ano, edituje se jiná součástka**

Úloha Kirchhoffovy zákony:

- b drží a edituje, a vkládá
- součástky mají bokem na hromádce po editaci je vkládají do obvodu

Participanti 10a,b (dvojice)

Úloha Ohmův zákon:

- a míří na schéma, b opisuje a počítá
- spolupracují při editaci, ovšem **nechávají součástku v editoru a vyskakuje na ně dialog**
- úspěšně vypočítali hodnoty, vše dosadili do obvodu a vyhodnotili

Úloha Kirchhoffovy zákony:

- neznají 2. Kirchhoffův zákon, po sestavení 1. rovnice s nápovědou 2. sestavili sami
- editace součástek v pořádku, výpočet správně, dosazení do obvodu OK

C.4 Dotazník po testování a jeho výsledky

Dotazník po testování – Elektrické obvody, virtuální učebnice

1. Co se vám na aplikaci líbilo? (způsob práce s aplikací, vzhled, atd.)

2. Co se vám na aplikaci nelíbilo? (způsob práce s aplikací, vzhled, atd.)

3. Ohodnoťte složitost zadání úloh 1- úplně jednoduché, 5 – velice složité:

    a. Ohmův zákon, zapojení rezistorů: 1 – 2 – 3 – 4 – 5

    b. Kirchhoffovy zákony: 1 – 2 – 3 – 4 – 5

4. Seřaďte následující úkony které jste prováděli při testování od nejjednodušších (1) po nejsložitější (5,6)?

A. Navigace mezi hlavním menu a ostatními obrazovkami

B. Pochopení zadání

C. Výpočet úloh

D. Editování součástek

E. Doplnění součástek do obvodu a jeho vyhodnocení

F. Jiné: .....

5. Ohodnoťte míru obtížnosti ovládání aplikace (manipulace s tabletem, el. součástkami, editorem, schématy) kde je 1 – velice snadná, 5 – velice obtížná.

1 – 2 – 3 – 4 – 5

6. Jak jste se cítil/a během řešení úlohy?

——— Pro dvojice ———

7. Jak se vám pracovalo ve dvojici?

8. Co se vám na práci ve dvojici líbilo?

9. Co se vám na práci ve dvojici nelíbilo?

Tabulka C.2: Výsledky dotazníku po testování (otázky pro všechny, P1 až P6 jsou participanti pracující jako jednotlivci, ostatní pracovali ve dvojicích)

	Co se vám na aplikaci líbilo?	**Co se vám na aplikaci nelíbilo?**	**Složitost úloh (OZ, KZ)**	**Úkony od nejjednodušších po nejsložitější**	**Míra obtížnosti ovládání aplikace**	**Jak jste se cítil?**
P1	chytré, pomůže, doplňování součástek	občasné chyby v aplikaci	3,3	A,B,D,E,C	2	hloupě (výpočet úlohy)
P2	3. rozměr, zapojování obvodu	editování součástek	2,4	A, (B,D,E), C	2	fajn, složitá druhá úloha
P3	3. rozměr	grafika	2,2	A,E,D,B,C	1	fajn, chtěl by více teorie zapomněl jí
P4	hezké, 3D	x	2,4	A,D,E,B,C	1	fajn
P5	moderní technologie, přehledné	problém s vyskakujícícm editorem součástek, po potvrzení změn	2,3	D,B,A,E,C	1	fajn
P6	grafika, moderní technologie, přesné	problém s vyskakujícícm editorem součástek, po potvrzení změn	2,4	E,A,D,B,C	2	fajn, ale složité zadání úlohy
P7a	3D	x	2,3	A,B, (D,E), C	1	fajn, ale nepamatuje si teorii
P7b	vzhled, moderní technologie, přesnost	problém s vyskakujícícm editorem součástek, po potvrzení změn	2,3	A,B,(D,E),C	1	vpoho

Tabulka C.2: Výsledky dotazníku po testování (otázky pro všechny, P1 až P6 jsou participanti pracující jako jednotlivci, ostatní pracovali ve dvojicích)

	Co se vám na aplikaci líbilo?	**Co se vám na aplikaci nelíbilo?**	**Složitost úloh (OZ, KZ)**	**Úkony od nejjednodušších po nejsložitější**	**Míra obtížnosti ovládání aplikace**	**Jak jste se cítil?**
P8a	3D	hodnota součástky špatně čitelná, bolí ho ruka, rozmazané?	2,3	A,D,E,B,C	1	dobrý
P8b	přehlednost, jednoduchost	x	1,3	E,D,A,B,C	2	nevěděl co čekat, nakonec dobré
P9a	3D, hezké	jednodušší je pouze spočítat příklad	2,3	A,E,D,B,C	2	nic moc, nechápe látku, zapomněl teorii
P9b	moderní technologie	hodnota součástky špatně čitelná, bolí ho ruka	2,4	A,D,E,B,C	2	nebaví ho fyzika
P10a	přehledné, pěkný vzhled, moderní, šikovné	x	2,3	A,D,E,C,B	1	dobře, fajn, pohodlně
P10b	úspora času, vzhled, přesnost, moderní	přidané papíry, lepší kdyby vše bylo v tabletu	3,4	A,D,B,E,C	1	dobře, fajn

Tabulka C.3: Výsledky dotazníku po testování (otázky pro dvojice)

	Jak se pracovalo ve dvojici?	**Co se vám líbilo na práci ve dvojici?**	**Co se vám nelíbilo na práci ve dvojici?**
P7a	je to jednodušší	právě spolupráce s kolegou	x
P7b	je to jednodušší	právě spolupráce s kolegou	x
P8a	dobře	kolega úlohu vypočítal	x
P8b	dobře	„víc hlav víc ví“	x
P9a	dobře	rozdělení práce	x
P9b	dobře	rozdělení práce, možnost konzultovat s kolegou	x
P10a	skvěle	spolupráce, zábava	x
P10b	chtěl do ruky tablet, ale zase mohl počítat a doplňovat do obvodu	spolupráce, zábava	kolega pracoval s tabletem

Příloha D

Testování předfinální verze aplikace

V této kapitole jsou zobrazeny dotazníky před a po výuce předfinální verze aplikace i se shrnutím jejich výsledků. Dále pak přepis poznámek vytvořených během pozorování výuky.

D.1 Dotazník před výukou a jeho výsledky

Dotazník před výukou – Elektrické obvody, virtuální učebnice

1. Vaše pohlaví: muž x žena

2. Váš věk:

3. Uveďte prosím Vaší poslední známku na vysvědčení z fyziky a matematiky:

   Fyzika:

   Matematika:

4. Jaké používáte mobilní zařízení (mobilní telefon, tablet, apod.), uveďte prosím i operační systém zařízení (Android, iOS, Windows Phone, apod)?

5. Co si představíte pod pojmem rozšířená realita?

6. Jaké konkrétní aplikace rozšířené reality znáte?

Tabulka D.1: Výsledky dotazníku před výukou

Pohlaví	**Věk**	**Prospěch fyzika**	**Prospěch matematika**	**Používaná mobilní zařízení**	**Pojem rozšířená realita**	**Aplikace rozšířené reality**
muž	17	4	4	Android smartphone	široká realita	široká aplikace
žena	17	1	1	Android smartphone	nějaké rozšíření mých znalostí reality rozšíření	x
žena	17	1	2	Android smartphone	x	x
žena	17	1	1	Android smartphone	jak je definována realita	x
žena	18	1	1	Android smartphone	x	x
muž	18	2	4	Android smartphone	elektronicky zmapovaná realita	x
muž	17	3	3	Symbian mobilní telefon	náhled na pozorovaný problém z více hledisek	x
muž	17	4	5	Android smartphone	expandovaná realita zahrnutí více hledisek do reality	Oculus rift
žena	17	3	2	Android smartphone	x	x
muž	18	2	2	iOS iPhone	nedokážu představit	x
muž	18	3	2	Android smartphone	Oculus rift	x
žena	18	2	2	Windows Phone smartphone, Android tablet	neznám	x
žena	18	4	4	Android smartphone, Android tablet	nevím	x
muž	17	4	2	Android smartphone	3D modely renderované přes výstup kamery	Nintento - AR Games
muž	17	4	3	iOS iPhone, Android tablet	neznám	x

Tabulka D.1: Výsledky dotazníku před výukou

Pohlaví	**Věk**	**Prospěch fyzika**	**Prospěch matematika**	**Používaná mobilní zařízení**	**Pojem rozšířená realita**	**Aplikace rozšířené reality**
žena	17	1	2	Android smartphone	x	x
žena	17	2	3	Android smartphone	x	x
žena	17	2	3	Android smartphone	x	x
žena	18	3	3	Android smartphone	x	x
žena	18	3	4	Android smartphone	x	x
žena	17	2	3	iOS smartphone	x	x
žena	17	2	3	Android smartphone	nové aplikace	x
žena	17	2	2	Android smartphone	x	x
žena	17	1	2	Android smartphone	x	x
muž	18	2	2	iOS smartphone	světová fakta	facebook
žena	17	3	3	iOS smartphone	x	x
muž	18	2	3	Android smartphone	x	facebook
muž	18	3	3	Android smartphone	média, sociální sítě	x
muž	17	2	3	Android smartphone	něco co spojí techniku s realným světem	facebook
muž	17	2	3	iOS smartphone	prvky pro zlepšení komunikace	facebook, twitter
žena	18	2	2	Android smartphone	x	x
muž	18	2	3	iOS smartphone	x	x
žena	17	3	3	iOS smartphone	na internetu seznamování s ostatními lidmi	facebook, instagram, youtube
žena	17	2	1	Android smartphone	mobilní aplikace	angry birds, whatsapp
žena	18	2	3	Android smartphone, iOS tablet	zobrazení reality foťákem nebo telefonem + objekty vytvořené počítačem	GPS, MovieMaker

Tabulka D.1: Výsledky dotazníku před výukou

Pohlaví	**Věk**	**Prospěch fyzika**	**Prospěch matematika**	**Používaná mobilní zařízení**	**Pojem rozšířená realita**	**Aplikace rozšířené reality**
žena	17	2	3	Android smartphone	nové aplikace, rozšíření reality o objekty počítačem	x
žena	17	1	2	Android smartphone	realita rozšířená o objekty přidané počítačem	3D hry
muž	18	1	2	iOS smartphone	realita s více možnostmi	FB
muž	18	2	2	Symbian mobilní telefon	augmentovaná realita, rozšířená pomocí el. zařízení	x

D.2 Poznámky pořízené při pozorování výuky

Níže jsou uvedeny stručné přepisy poznámek pořízených během obou vyučovacích bloků.

D.2.1 Výuka 1. blok, kapitola „Ohmův zákon, zapojení rezistorů“ - 15 studentů

- Volt-ampérová charakteristika - několik studentů neví kam má kliknout do tabulky pro doplnění vlastního materiálu
- Volt-ampérová charakteristika - v některých případech nevyjíždí s layout s vysunutím klávesnice
- Interaktivní úloha s rezistory - po smazání celé hodnoty a opuštění TextView se nelze vrátit
- Rozšířená realita - jedné dvojici neostří fotoaparát tabletu na editor, stíní si

D.2.2 Výuka 2. blok, kapitola „Kirchhoffovy zákony“ - 24 studentů

- Rozšířená realita - v zadání první úlohy v rozšířené realitě je vypočtená hodnota součtu rezistorů na špatném místě

D.3 Dotazník po výuce a jeho výsledky

Dotazník po výuce – Elektrické obvody, virtuální učebnice

1. Co se vám na aplikaci líbilo? (způsob práce s aplikací, vzhled, atd.)

2. Co se vám na aplikaci nelíbilo? (způsob práce s aplikací, vzhled, atd.)

3. Ohodnoťte míru obtížnosti ovládání aplikace (manipulace s tabletem, el. součástkami, editorem, schématy) kde je 1 – velice snadná, 5 – velice obtížná.

1 – 2 – 3 – 4 – 5

4. Ohodnoťte, jak se Vám pracovalo s editorem součástek kde je 1 – velmi dobře, 5 – velmi špatně. v případě známky 3 a vyšší uveďte prosím, co Vám na práci s editorem vadilo.

1 – 2 – 3 – 4 – 5

5. Použil/a jste někdy během řešení úloh v rozšířené realitě nápovědu? Pokud ano byla pro Vás užitečná?

6. Jak jste se cítil/a během řešení úloh v rozšířené realitě?

7. Jak se vám líbily interaktivní úlohy mimo rozšířenou realitu (VA charakteristika, doplnění pyramidy Ohmova zákona, ...)? Která se vám líbila nejvíce, a která naopak nejméně?

8. Pokud jste pracovali ve dvojici, jak se vám v ní pracovalo, co se vám na práci ve dvojici líbilo, případně nelíbilo?

Tabulka D.2: Výsledky dotazníku po výuce

Co se vám na aplikaci líbilo?	**Co se vám na aplikaci nelíbilo?**	**Ovládání**	**Hodnocení editor + poznámka**	**Nápověda + užitečnost**	**Jak jste se cítil?**	**Interaktivní úlohy mimo AR**	**Práce ve dvojici,hodnocení**
lehká manipulace, pěkný vzhled, jasné, vypadá prakticky	x	1	3 - neostřil na editor	nepoužili	příjemně, bavilo, objasnilo spoustu věcí	vše se líbilo	příjemné, pohodlné, OK
x	chyběl tutoriál jak použít AR	2	3 - špatný, neostřil	ano, ano	:)	:)	:)
dobré procvičení	dlouhé načítání modelů	2	3 - neostřil nechytal	ano, ano	dobře	líbily	líbilo
vzhled, 3D modely	editace součástek	2	4 - neostřil nechytal	nepoužili	moc mě to nebavilo	jednoduché, srozumitelné, líbili se stejně	dobré
názorné představení, jednoduchá manipulace	x	1	1	nepoužili	konečně jsem měla pocit že tomu rozumím	vše se líbilo, hezky vysvětlené	dobře, vystřídali jsme se, vše jsme si vyzkoušely
vše funkční, rychlé reakce, hezké zpracování	x	1	1	nepoužili	x	vše zajímavé doplnění teorie	spolupracovalo se dobře
3D modely, vzhled	dlouhé nahrávání modelů	2	2	ano, ani ne	normálně	více méně ano	dobře
jednoduché ovládání přehledné	x	2	1	ano, pro zjištění postupu ano	povznešeně	užitečné opakování	příjemně
nezajímá mě to k ničemu	nezajímá mě to, k ničemu	1	1	x	x	x	x

Tabulka D.2: Výsledky dotazníku po výuce

Co se vám na aplikaci líbilo?	**Co se vám na aplikaci nelíbilo?**	**Ovládání**	**Hodnocení editor + poznámka**	**Nápověda + užitečnost**	**Jak jste se cítil?**	**Interaktivní úlohy mimo AR**	**Práce ve dvojici,hodnocení**
3D modely	x	2	2	ano, ano	hladově	vše se líbilo	vzhled spolupracovníka se mi nelíbil
zábavné, provedení 3D modely	dlouhé načítání modelů	2	1	ano, ano	v pořádku	víc se líbila AR	nemohl jsem kolegyni upravit pomocí editoru
jednoduché ovládání, funkční	x	1	2	ano, ano	dobře	víc se líbila AR	x
vzhled, čitelné, pěkné, vyhovující způsob práce z aplikací	x	2	2	ano, ano	rozšířeně, neobvyklý zážitek	zajímavé, nejvíce se líbila pyramida	x
vzhled, pěkné, jednoduché, bezproblémová práce	x	1	1	ano, ano	něco nového	líbily se, nejvíce pyramida	x
způsob práce	x	2	2	ano, ano	nijak výjimečně, ale zajímavé	x	x
snímání fotoaparátem, vzhled rozšířené reality	x	2	1	nepoužili	celkem sranda	x	nepůjčil mi to
rozšířená realita, pěkný vzhled	x	1	1	nepoužili	úchvatně	x	nepůjčil mi to
3. rozměr, pěkné reakce	x	2	1	ano, ano	x	rozšířená realita se líbila více	x

Tabulka D.2: Výsledky dotazníku po výuce

Co se vám na aplikaci líbilo?	**Co se vám na aplikaci nelíbilo?**	**Ovládání**	**Hodnocení editor + poznámka**	**Nápověda + užitečnost**	**Jak jste se cítil?**	**Interaktivní úlohy mimo AR**	**Práce ve dvojici,hodnocení**
pěkná grafika, atraktivní	x	2	2	ano,ano	x	x	x
vzhled aplikace 3D	x	2	2	ano,ano	trochu zmateně (nevím jak spočítat)	pyramida	ve dvou trochu hůře vidět 3D, jinak fajn
vzhled	mazání rovnic 1.KZ, 2.KZ	1	2	nepoužili	tak různě	pyramida	byla to sranda
rozšířená realita, 3D, interaktivní úlohy	x	2	2	ano, ano	zajímavé, líbilo se, chtěla bych si to zkusit znovu	pyramida	dobré, můžeme si poradit, zvážit možnosti
práce s aplikací, vizualizace objektů	x	1	1	ano, ano	3D mě nadchlo	jednoduchá a logická forma úloh	znamenitě
vizualizace, 3D, jednoduché ovládání	x	1	2	ano, ano	nadšená, super 1. zkušenost s rozšířenou realitou	x	super spolupráce, žádný problém
x	x	2	2	ano, ani ne	zvláštně	x	x
pěkná zkušenost, 3D	při vypnutí aplikace se mi do scénáře vygenerovaly nové hodnoty	3	2	nepoužili	skvěle	pyramida	ano, dobrá nálada ve dvojici
vzhled, práce z aplikací	x	2	2	ano, ano	velmi dobře	líbili se	ano, kolega doplnil moje neznalosti

Tabulka D.2: Výsledky dotazníku po výuce

Co se vám na aplikaci líbilo?	**Co se vám na aplikaci nelíbilo?**	**Ovládání**	**Hodnocení editor + poznámka**	**Nápověda + užitečnost**	**Jak jste se cítil?**	**Interaktivní úlohy mimo AR**	**Práce ve dvojici,hodnocení**
přehlednost, vzhled, nápad	x	1	1	ano, ano	nevím?	pyramida fajn, KZ trochu obtížné	vše ve spolupráci s kolegyní super
přehlednost	x	2	2	ano, ano	překvapeně	pyramida fajn, KZ trochu obtížné	pracovalo se dobře, vše bylo fajn
způsob práce s aplikací, vzhled	x	2	1	nepoužili	krásně	KZ se líbilo	kolega je blázen
způsob práce s aplikací, vzhled	x	2	2	nepoužili	krásně	KZ se líbilo	kolega mi nepomáhal
pěkný nápad, chytrá aplikace	práce s aplikací	3	3 (nešlo vložit desetinné číslo)	nepoužili	jak v imagináriu	pyramida fajn, KZ trochu obtížné	doplňovali jsme se při tvorbě schémat
grafika	obtížná úloha (KZ)	4	4 (nepřehlednost)	ano, ani ne	divně	pyramida fajn	lepší než pracovat sám
zpracování, jednoduchost, rozšířená realita	x	3	2	nepoužili	komfortně	pyramida fajn	spolupráce dobrá, jiné názory se hodí
vše	x	2	1	ano,ano	zajímavé	VA charakteristika	líbilo
grafika	nutnost editace součástek a počítání	2	2	nepoužili	skvěle	x	soused nic nevěděl
práce s tabletem	pomalé načítání modelů	3	2	nepoužili	dobře	x	soused nic nevěděl

Tabulka D.2: Výsledky dotazníku po výuce

Co se vám na aplikaci líbilo?	**Co se vám na aplikaci nelíbilo?**	**Ovládání**	**Hodnocení editor + poznámka**	**Nápověda + užitečnost**	**Jak jste se cítil?**	**Interaktivní úlohy mimo AR**	**Práce ve dvojici,hodnocení**
vzhled	práce s učitelem	3	2	nepoužili	chytře	Ohmův zákon, rozšířená realita ale lepší	dobře
vzhled	x	4	2	ano,ano	chytře	pyramida	dobře