

# Posudek bakalářské práce

Matematicko-fyzikální fakulta Univerzity Karlovy

**Autor práce** Vojtěch Antošík  
**Název práce** Modul pro detekci objektů ve video streamu v reálném čase  
**Rok odevzdání** 2020  
**Studijní program** Informatika      **Studijní obor** Programování a softwarové systémy

**Autor posudku** RNDr. Martin Kruliš, Ph.D.      **Role** Oponent  
**Pracoviště** Katedra softwarového inženýrství

Prosím vyplňte hodnocení křížkem u každého kritéria. Hodnocení *OK* označuje práci, která kritérium vhodným způsobem splňuje. Hodnocení *lepší* a *horší* označují splnění nad a pod rámec obvyklý pro bakalářskou práci, hodnocení *nevyhovuje* označuje práci, která by neměla být obhájena. Hodnocení v případě potřeby doplňte komentářem. Komentář prosím doplňte všude, kde je hodnocení jiné než *OK*.

<b>K celé práci</b>	lepší	OK	horší	nevyhovuje
Obtížnost zadání	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Splnění zadání	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Rozsah práce ... <i>textová i implementační část, zohlednění náročnosti</i>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>

**Komentář** Cílem práce bylo navrhnout a implementovat modul pro detekci objektů ve videu v reálném čase, který by byl založen na již existujícím natrénovaném modelu SSD. Autor pouze vzal daný modul pro statickou detekci a přímočaře jej aplikoval na jednotlivé snímky z videa. Hlavním přínosem práce bylo pouze sestavení již existujících komponent a knihoven (dekódování videa, detektor, knihovny pro práci s databází) a jejich sestavení do pipeline. Autor implementoval zcela sám pouze modul pro konverzi formátu barev, což je přímá aplikace triviálních vzorců. Nejobtížnější částí práce bylo zajistit interoperabilitu mezi modulem v Pythonu a pipeline v C++ (nutno podotknout, že C++ si vybral autor sám z ne zcela zjevných důvodů) a implementace porpory CUDA kernelů v pipeline a předávání dat mezi nimi (bez nutnosti opakovaného přenosu mezi hlavní pamětí a GPU).

Cílem práce bylo také provést profilování a navrhnout optimalizace. Profilování bylo provedeno poměrně minimalistickým způsobem pouhým změřením propustnosti jednotlivých modulů na jednom konkrétním kusu hardware. Autor žádné zásadní optimalizace nenavrhuje ani neposkytuje vysvětlení pro některé pozorované výkonnostní nedostatky.

Celkově je práce svým rozsahem a mírou splnění zadání na hranici obahjitelnosti, nicméně zásadní nedostatky rozebrané v dalších částech posudku ji za tuto hranici posouvají.

**Textová část práce**

	lepší	OK	horší	nevyhovuje
Formální úprava ... <i>jazyková úroveň, typografická úroveň, citace</i>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Struktura textu ... <i>kontext, cíle, analýza, návrh, vyhodnocení, úroveň detailu</i>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Analýza	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Vývojová dokumentace	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Uživatelská dokumentace	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

**Komentář** V první části práce autor věnuje jednu kaptolu na úvod do problematiky. V této kapitole se rozebírají pouze dvě věci – reprezentace rastrových obrazových dat v různých barevných prostorech (což je zde poněkud zbytečně podrobné a vytržené z kontextu, zejména když autor nepopisuje detaily reprezentace videa) a detekce objektů v obrazu (kde autor pouze směřuje k rychlému obhájení použití detektoru SSD, což jednak není příliš přesvědčivé a jednak zbytečné, protože SSD detektor byl určen v zadání). Naopak zde zcela postrádám detailnější rozbor fungování SSD, bez kterého je pak např. nemožné porozumět spouštěcím parametrům přiložené aplikace.

Analýza práce rozebírá nutnost paralelizace pro plnohodnotné využití současného hardware (známý fakt) a obhajuje pipeline jako vhodnější architekturu (oproti monolitickému kódu) pro zpracování videa (známý fakt). Není proveden žádný rozbor existujících systémů, které řeší podobné problémy (např. pipeline pro kompresi videa), nebo alespoň obecných paralelních frameworků, které mají pipeline již implementovanou (např. TBB). Rovněž zcela chybí analýza vhodnosti použitých knihoven (pro dekodování videa), existujících knihoven pro konverzi barev pomocí CUDA (autor si implementuje sám), nebo propojení s databází (čtenář se ani nedozví, že se jedná o relační databázi). Výběr samotného jazyka (kde je uvažován pouze Python a C++) je rovněž podivně odůvodněn – SSD je napsán v Pythonu, takže výběr C++ výrazně zesložití práci, neboť je potřeba zajistit interoperabilitu mezi SSD a ostatními moduly pipeline.

Experimentální vyhodnocení propustnosti pipeline je provedeno korektně, ale minimalisticky. Některé výsledky nejsou prezentovány zcela srozumitelně (zejména obrázek 4.1, kde není ani uvedeno, co jednotlivé barvy znamenají). Zcela postrádám detailnější analýzu profilování (co přesně brzdí vyhodnocování v jednotlivých modulech) a celkové srovnání s jakoukoli alternativou (např. alespoň s monolitickou implementací, která je jako jediná alternativa uvažována v rozboru). Dále postrádám jakékoli výsledky z reálného nasazení, přinejmenším diskusi nad tím, zda autor vyzkoušel pustit svůj prototyp nad reálnými daty a zda skutečně detekce fungovala v reálném čase.

Uživatelská dokumentace obsahuje návod na sestavení a přehled spouštěcích argumentů. Hlavním nedostatkem je zde fakt, že samotné argumenty se týkají převážně konfigurace SSD modulu, který není v práci popsán, takže čtenář práce nemá jak pochopit jejich význam. Parametr určující váhy modelu navíc vyžaduje vstupní soubor, jehož formát není nikde popsán a není ani řečeno, jak takový soubor vyrobit.

Programátorská dokumentace je poněkud stručnější, ale kompletní. Bylo by vhodné ji doplnit ještě o přehled, jak je organizován zdrojový kód, i když vzhledem k velmi malému rozsahu projektu není absence přehledu kritická. Dále by bylo vhodné uvést konkrétní příklad, jak vyrobit nový modul a začlenit ho do pipeline.

Po jazykové a typografické stránce je práce na velmi dobré úrovni. Citace jsou korektní, ale je jich malý počet (důsledek nedostatečně provedené rešerše a analýzy).

**Implementační část práce**

lepší OK horší nevyhovuje

Kvalita návrhu ... architektura, struktury a algoritmy, použité technologie	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Kvalita zpracování ... jmenné konvence, formátování, komentáře, testování	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Stabilita implementace	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>

**Komentář** K práci nebyla odevzdána kompletní příloha všech zdrojových kódů potřebných k sestavení korektního docker image (ssd detektor modul), ani není v práci uveden link na repozitář, kde by potřebné soubory byly ke stažení. Po dodání souborů mailem se podařilo docker image vytvořit a aplikaci spustit, avšak již při výchozím nastavení parametrů dodje k běhové chybě. Vzhledem k absenci jakýchkoli testovacích dat (i video jsem byl nucen pořídit vlastní) a vzhledem k naprosto nedostatečnému popisu parametrů v uživatelské dokumentaci jsem byl nucen testování zanechat. Další pokus byl otestovat alespoň demo této implementace, které běží jako webová aplikace na serveru videolytics, avšak ani toto demo se mi nepodařilo zprovoznit.

Většina kódu je napsána v C++ (případně v CUDA) vyjma wrapperu na SSD modul, který je napsán v Pythonu. Struktura a dekompozice kódu je průměrná, celkově obvyklá pro vybraný jazyk. Komentáře jsou velmi sporadické, ale díky rozumným jmenným konvencím je kód relativně čitelný.

Největším nedostatkem je návrh a implementace samotné pipeline. Autor používá základní paralelní primitiva (thready, mutexy) a to velmi hrubým způsobem, který může vést k řadě výkonnostních problémů (např. každý modul pipeline má vlastní vlákno, čímž je omezena škálovatelnost systému počtem modulů, jednotlivá vlákna blokují, pokud dojde k zahlcení fronty atd.). Na jednu stranu by sice bylo možné toto řešení akceptovat jako nedokonalé, ale víceméně funkční. Na druhou stranu by se mělo jednat o hlavní přínos této práce a důraz na rychlé zpracování je několikrát zopakován v textu, takže nelze než hodnotit tento cíl jako ne zcela vyřešený.

V kódu se také objevuje řada drobnějších nedostatků, které zde není prostor vyjmenovat. Jako příklady bych uvedl zadrátování konfiguračních parametrů pro připojení k databázi (včetně hesla!) přímo do kódu, opakovaná alokace GPU paměti, nepříliš vhodné použití shared pointerů a raw pointerů, nebo switch-case konstrukce s částečně opakujícím se kódem.

Kladně lze hodnotit použití moderních nástrojů pro sestavení a deployment (cmake, docker), na druhou stranu není jasné, zda byl pro vývoj použit systém pro správu verzí (práce nemá odkaz na veřejnou repozitář, přičemž vzhledem k povaze práce by to bylo očekáváno).

**Celkové hodnocení** Neprospěl(a)

**Práci navrhuji na zvláštní ocenění** Ne

**Datum** 30. června 2020

**Podpis**