



MATEMATICKO-FYZIKÁLNÍ FAKULTA Univerzita Karlova

BAKALÁŘSKÁ PRÁCE

Matěj Ščerba

Analýza videozáznamu skoku o tyči

Katedra softwarového inženýrství

Vedoucí bakalářské práce: Ing. Michal Bartoš, Ph.D.

Studijní program: Informatika (B1801)

Studijní obor: IOI (1801R008)

Praha 2021

Prohlašuji, že jsem tuto bakalářskou práci vypracoval(a) samostatně a výhradně s použitím citovaných pramenů, literatury a dalších odborných zdrojů. Tato práce nebyla využita k získání jiného nebo stejného titulu.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona v platném znění, zejména skutečnost, že Univerzita Karlova má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle §60 odst. 1 autorského zákona.

V dne
Podpis autora

Poděkování.

Název práce: Analýza videozáznamu skoku o tyči

Autor: Matěj Ščerba

Katedra: Katedra softwarového inženýrství

Vedoucí bakalářské práce: Ing. Michal Bartoš, Ph.D., Katedra softwarového inženýrství

Abstrakt: Skok o tyči je jedna z technicky náročnějších lehkoatletických disciplín, proto se analýzou skoků na většině mezinárodních závodů zabývají vědci zkoumající pohyb lidského těla. Pro trenéry a závodníky by bylo vhodné, aby měli k dispozici podobné prostředky i při trénincích, aniž by museli investovat do specializované techniky. Má práce tuto problematiku řeší. S použitím počítače a kamery lze z jediného videozáznamu získat model atleta a zároveň analyzovat parametry jím provedeného skoku. Výsledné parametry se posléze zanesou do grafů a jejich hodnoty lze zkoumat i v různých fázích skoku za použití prohlížeče videozáznamu.

Klíčová slova: skok o tyči detekce pozice těla videozáznam

Title: Analysis of pole-vault video recording

Author: Matěj Ščerba

Department: Department of Software Engineering

Supervisor: Ing. Michal Bartoš, Ph.D., Department of Software Engineering

Abstract: Abstract.

Keywords: pole vault human pose detection video recording

Obsah

Úvod	3
0.1 Související práce	4
1 Teorie	5
1.1 Skok o tyči	5
1.1.1 Analýza technického provedení skoku	5
1.1.2 Analýza biomechanických parametrů skoku	6
1.2 Zpracování obrazu	8
1.2.1 Detekce člověka	8
1.2.2 Tracker	8
1.2.3 Detekce kostry	8
2 Návrh řešení	9
2.1 Souřadné systémy	9
2.1.1 Snímek	9
2.1.2 Video	9
2.1.3 Prostor	9
2.2 Vstup programu	10
2.3 Analýza videa	10
2.3.1 Zpracování videa	10
2.3.2 Nalezení atleta	11
2.3.3 Detekce kostry	12
2.3.4 Konverze kostry do 3D modelu	14
2.3.5 Analýza parametrů	14
2.4 Analýza modelu	17
2.5 Výstup programu	17
2.5.1 Uložení výstupu	17
2.5.2 Zobrazení výstupu uživateli	18
2.6 Programové vybavení	18
2.6.1 Počítačové vidění a zpracování obrazu	18
2.6.2 Grafická reprezentace výstupu	19
3 Realizace	20
3.1 Zvolené prostředky	20
3.2 Konverze souřadných systémů	20
3.3 Struktura programu	21
3.4 Vstup programu	22
3.5 Analýza videa	22
3.5.1 Zpracování videa	22
3.5.2 Nalezení atleta	22
3.5.3 Detekce kostry	24
3.5.4 Konverze kostry do 3D modelu	28
3.5.5 Analýza parametrů	29
3.6 Analýza modelu	32
3.7 Výstup programu	32

3.7.1	Uložení výstupu	32
3.7.2	Zobrazení výstupu uživateli	32
4	Experimenty	34
4.1	Volba úvodní pozice atleta	34
4.1.1	Pozice ohraničujícího rámečku	34
4.1.2	Velikost ohraničujícího rámečku	34
4.2	Pohyb kamery	34
4.2.1	Vertikální pohyb kamery	34
4.2.2	Rotace kamery	34
4.3	Pozice kamery	34
4.3.1	Směr rozběhu	34
4.3.2	Úhel záběru	34
4.3.3	Vzdálenost kamery	34
4.4	Další postavy ve videu	34
4.5	Frekvence snímků	34
4.6	Závody	35
4.7	Shrnutí	35
Závěr		36
Seznam použité literatury		37
Seznam obrázků		38
Seznam tabulek		39
Seznam použitých zkratek		40
A Přílohy		41
A.1	První příloha	41

Úvod

Při trénincích a soutěžích ve skoku o tyči využívají atleti spolu s trenéry videozáznamy provedených skoků. Videozáznam poté využívají pro detekci chyb v technickém provedení skoku a jejich případný vliv na výsledný výkon. Pro výkon při skoku o tyči jsou kromě techniky důležité také biomechanické parametry skoku. Mezi tyto parametry patří například rychlosť atleta při rozběhu.

V tréninkovém procesu je důležité sledovat výkonnost atleta. K tomuto účelu slouží především měření času běhu různých vzdáleností. Pro skok o tyči je vhodné aplikovat podobné metody právě při skokanských trénincích. Jednou z možností, jak toho docílit, je použití specializované techniky pro měření rychlosti atleta při rozběhu. Mezi specializovanou techniku lze zařadit radary nebo fotobuňky. Za použití vhodně rozmístěných fotobuněk lze poměrně přesně určit rychlosť v různých částech rozběhu. Využití těchto metod je finančně náročné a jediným parametrem, který lze sledovat je pouze rozběhová rychlosť atleta.

S rozvojem počítačového vidění přibývá na světových atletických soutěžích studií zabývajících se pohyby sportovců (Athletics, 2021). Výsledky těchto studií jsou pro atlety a jejich trenéry vynikající zpětnou vazbou. Tyto studie využívají několik kamer k zachycení pohybu atleta, kamery nejprve kalibrují za použití přesně změřených konstrukcí s vyznačenými body. Pro analýzu pohybu poté využívají 3D model atleta, který z videí získají. Výsledný model zachycuje polohu kloubů těla a na základě jejich pohybů lze jeho pohyb analyzovat. Takovýto model je možné použít k analýze mnohem více parametrů skoku o tyči než jen rychlosť běhu atleta.

Studie se většinou zabývají porovnáváním pohybu různých atletů na základě získaných parametrů. V tréninkovém prostředí je zvykem využívat pouze videozáznamy a případné porovnání skoků probíhá spuštěním záznamů, které dané skoky zachycují, následným rozborem techniky jednotlivých skoků a hledáním případných rozdílů. Tato forma porovnání skoků ovšem nebude v potaz biomechanické aspekty pohybu atleta, které mohou techniku skoku ovlivnit. Hodnoty parametrů, kterými se specializované studie zabývají, lze z videa odhadnout pro účely porovnání skoků, ale ani zkušení trenéři nejsou schopni přesně určit jejich hodnoty.

Podobné studie pohybu jsou užitečným tréninkovým prostředkem, ovšem jejich využití jen v malém počtu tréninků se nejeví jako reálné. Dostupnějším přístupem k tomuto problému by byla aplikace, běžící na mobilním zařízení, která by analyzovala pohyb atleta na základě modelu získaného z jediného videa. Výsledkem této analýzy by byly graficky znázorněné parametry skoku a případné porovnání s hodnotami parametrů jiných skoků. Takováto aplikace by výrazně přispěla k přesnější analýze tréninkových a závodních skoků mimo soutěže světové úrovni.

Má práce představuje základ pro vznik podobné aplikace. Mým cílem je extrakce modelu atleta z jediného videozáznamu, jeho následná analýza pro zisk hodnot příslušných parametrů a následné zobrazení hodnot parametrů jejich zanesením do grafů.

Text jsem rozdělil do tří kapitol. V první se zabývám teorií. V této kapitole popisuji skok o tyči, způsoby jeho analýzy, možnosti využití videozáznamů a důle-

žité biomechanické parametry skoku. Dále se ve stejné kapitole zabývám teorií počítačového vidění, konkrétně detekcí člověka, způsobem fungování trackerů a detekcí kostry člověka.

Druhá kapitola se věnuje realizaci programu. V úvodu této kapitoly se věnuji pracem zabývajícím se související tematikou, následně rešerši dostupných knihoven k realizaci programu, stručnému návrhu řešení a samotné realizaci. V poslední zmíněné části se věnuji podrobnému popisu programu.

Ve třetí a zároveň poslední kapitole se věnuji experimentům, konkrétně zisku testovacích dat a rozboru funkcionality programu na různých videích.

0.1 Související práce

1. Teorie

1.1 Skok o tyči

Skok o tyči je jedna z technicky náročných atletických disciplín. Atlet se nejprve s tyčí rozběhne po rozběžisti, následně zasune tyč do kastlíku, poté se odrazí, provede skok a dopadne do doskočiště. Kastlík je místo v zemi, kam atlet při rozběhu zasune tyč, aby měl při skoku stabilní oporu.

Atlet se při skoku pohybuje vzhůru za pomocí narovnávající se tyče, kterou ohne s použitím energie získané při rozběhu. Tyče se liší délkou a tvrdostí, tvrdší tyč je náročnější ohnout, aby atleta vynesla přes latku, ale akumulovaná energie tyče je vyšší, a tedy vyúsťí ve vyšší skok.

Pro pozici těla atleta při rozběhové fázi je typický značně omezený pohyb paží a mírné natočení trupu. Důvodem tohoto nestandardního způsobu běhu je skutečnost, že atlet nese tyč, a přesto se snaží vyvinout maximální kontrolovanou rychlosť. Rychlosti, které atlet dosahuje v momentu odrazu, se říká náběhová rychlosť. Při skoku o tyči (a všech ostatních skokanských disciplínách) má atlet vyznačené místo, ve kterém začíná svůj pokus. Toto místo si volí sám, vliv na jeho polohu má především fyzická zdatnost atleta a počet kroků rozběhu. Počet kroků závodního rozběhu atletů světové úrovně se pohybuje v rozmezí 16 až 20 kroků, tedy přibližně 35 až 45 metrů od zadní hrany kastlíku. Atlet se rozbíhá s tyčí ve vzduchu, postupně ji spouští, dokud není rovnoběžně se zemí. Následně provádí zásun - pohyb, při němž zasune tyč do kastlíku a dostane paže nad hlavu. V pozici s pažemi nad hlavou se atlet odráží a přenáší energii do tyče. Na charakter skoku má vliv také místo odrazu. Následný skok lze rozdělit do několika částí.

První z nich je odraz, při němž je kladen důraz na efektivitu přenosu energie získané při rozběhu do tyče. Po této krátké fázi skoku následuje zvrat. Jedná se o pohyb způsobený švihnutím odrazové nohy a paže, která se tyče drží výše, proti sobě - směrem dopředu. Po provedení zvratu se atlet dostane do pozice vzhůru nohama. V této pozici atlet provádí obrat, při němž se otáčí o 180 stupňů podél vertikální osy, aby byl čelem k latce. Následuje odraz od tyče a přechod latky, v této fázi se atlet snaží dostat boky co nejvýš a zajít tak plynulý skok přes latku. Latku atlet překonává nohama napřed, čelem k latce.

1.1.1 Analýza technického provedení skoku

Způsob provedení skoku je podstatným ukazatelem pro výsledný výkon. Jen drobná změna v jediné fázi pokusu může dramaticky ovlivnit charakter a výšku celého skoku. Tím pádem mají skokané o tyči pro nácvik techniky vyhrazeno několik tréninků týdně. Jedná se o skokanské tréninky a tréniny zabývající se nácvikem techniky prostřednictvím gymnastických prvků simulujících pohyb atleta na tyči. Pro mou práci jsou důležitější skokanské tréninky.

Skokanské tréninky začínají podobně, jako ostatní atletické tréninky, tedy rozcvičením. Po klasickém rozcvičení následuje příprava na samotné skákání, tato příprava se u jednotlivých atletů může lišit. Mnoho atletů před samotným skákáním provede několik cvičných zásunů a rozběhů s tyčí. Následují skoky z krátkého rozběhu, standardně se jedná o 6 až 8 kroků, někteří atleti provádí tyto

skoky bez ohýbání tyče. Již na základě těchto cviků lze určit, na jaké prvky skoku by se měl atlet zaměřit. Po této fázi následuje skákání z dlouhého rozběhu. Počet kroků se liší podle fáze sezony, ve které se atlet nachází. V závodním období jsou typické rozběhy delší, v přípravném období kratší, jelikož tréninky v přípravném období jsou zpravidla fyzicky náročnější.

Při tréninku se opakuje následující situace. Atlet provede skok a následně ho konzultuje s trenérem. Předmětem konzultace je především technické provedení skoku. S rozvojem moderních technologií jsou konzultace ve většině případů doprovázeny sledováním a rozborem videozáznamu zahycujícího právě provedený skok. S pomocí tohoto videozáznamu lze přesně určit místo odrazu a pohyb atleta při rozběhu a následném skoku. Na pořízeném videozáznamu lze spolehlivě detekovat technické nedostatky skoků, které by se atlet měl v následujících pokusech snažit eliminovat.

Videozánam je nejčastěji pořizován na mobilní telefon, případně tablet. Pozice kamery se nejčastěji nachází na kolmici k rozběžišti, která prochází místem odrazu. Z tohoto místa je poměrně dobře vidět jak rozběh, tak skok. Navíc se takto s velikou přeností dá určit místo odrazu a pozice atletova těla při přenosu energie do tyče při odrazu. Strana, z které je skok natočen se často mění, zálaží na prvku skoku, který má trenér s atletelem v plánu zkoumat.

Závody probíhají podobně, po každém skoku opět dochází ke konzultaci, ovšem na mezinárodních závodech není běžné, že by atlet viděl záznam skoku, který trenér pořídil, na většině soutěží je to zakázané pravidly. Trenér tedy pouze popisuje nedostatky skoku a probíhá diskuse s atletelem ohledně zaměřující se následujícím postupem. Při závodech není kladen takový důraz na změny v technice jako při tréninku, typicky probíhá jen rozborek detailů nebo posouvání místa odrazu pro optimální přenos energie do tyče.

1.1.2 Analýza biomechanických parametrů skoku

S rozvojem moderní techniky přibývá studií zabývajících se pohybem těla atleta při skoku o tyči. Podkladem pro tyto studie jsou především mezinárodní závody, případně mistrovství republiky. Příkladem je biomechanická zpráva z finále mužů Mistrovství světa v atletice 2017 (Gravestock a kol., 2017), na níž se podíleli pracovníci Leeds Beckett University.

K analýze parametrů se využívá několik kamer, jejichž záznam slouží jako podklad k vytvoření modelu atleta v průběhu celého skoku. Tento model je následně analyzován pro zisk konkrétních parametrů skoku. Podobné studie se nejčastěji zabývají následujícími parametry:

Délka jednotlivých kroků rozběhu

Na efektivitu přenosu energie do tyče má vliv délka kroků na konci rozběhu. U většiny atletů je poslední krok rozběhu kratší oproti předešlým krokům. Rozdíl v délce předposledního a posledního kroku se může pohybovat v řádech desítek centimetrů. Důvodem pro zkrácení posledního kroku je sešlápnutí posledního kroku pod tělo atleta, tím atlet ztratí méně horizontální rychlosti při odrazu. Zkrácení posledního kroku doprovází odraz pod menším úhlem, což pro skok o tyči není tak velký problém jako pro skokany do dálky.

Doba oporové fáze kroku

Technika běhu je značně ovlivněna dobou oporové fáze jednotlivých kroků. Doba oporové fáze souvisí s rychlostí běhu, mírou pokrčení stojné nohy a výškou boků a kolen. Z prostého videozáznamu se složitě určuje doba oporové fáze kroku, a tudíž se při této analýze věnuje zvýšená pozornost právě míře pokrčení stojné nohy a výše boků a kolen.

Doba oporové fáze kroku a délka kroku se kromě analýzy videozáznamu dá měřit speciálními pásy, které se položí na zem po obou stranách rozběžště.

Náběhová rychlosť

Náběhová rychlosť má značný vliv na přenos energie rozběhu do tyče, rychlejší atleti tedy zpravidla používají tvrdší a delší tyče, které umožňují vyšší skoky.

Náběhovou rychlosť lze měřit několika způsoby. Pro tréninkové účely se nejčastěji používají rovnoměrně rozmístěné fotobuňky, z doby běhu mezi nimi lze snadno vypočítat průměrnou rychlosť v daném úseku. Tuto metodu používají především sprinteré, pro skok o tyči není příliš vhodná, rychlosť se v momentu odrazu dramaticky mění a tyč může ovlivnit detekci fotobuňek. Následující možností je použití radaru, s kterým lze zanést aktuální rychlosť do grafu. Při snímání atleta ze zadu téměř nedochází k chybám měření. Způsob měření náběhové rychlosti použitý ve výše zmíněné studii (Gravestock a kol., 2017) spočívá v analýze detekovaného modelu v souřadném systému, který umožňuje výpočet rychlosti jednotlivých částí těla ve standardních jednotkách, typicky se uvádí v ms^{-1} .

Výška boků v průběhu rozběhu

V průběhu rozběhu je zajímavé pozorovat výšku boků, výrazné výkyvy výšky boků mají za následek výkyvy rychlosti. Plynulost rozběhu tedy koreluje s plynulostí výšky boků v průběhu rozběhu. Pro rovný skok je důležité, aby se atlet rozeběhl kontrolovaně, což se může projevit ve výšce boků, výkyvy ve výšce boků také souvisí s proměnlivou délkou kroků a dobou oporové fáze kroku. Nejdůležitější moment pro zkoumání výšky boků jsou poslední kroky rozběhu.

Místo odrazu

Vzdálenost místa odrazu od zadní hrany kastlíku je podstatným ukazatelem efektivity přenosu energie do tyče. Je vhodné, aby měl atlet v místě odrazu horní paži přímo nad sebou. Vhodná vzdálenost místa odrazu je individuální, někteří atleti preferují odraz blíže ke kastlíku než jiní. Pozice odrazu se odvíjí od technického provedení skoku, výšky a odrazových schopností atleta.

Výška úchopu

Hodnota tohoto parametru souvisí s délkou tyče a odvíjí se od ní místo odrazu. Může se uvádět jako vzdálenost úchopu horní ruky od spodního konce tyče nebo jako výška úchopu horní ruky nad zemí, když je tyč svislá v kastlíku. Kastlík je hluboký 20 cm, tedy hodnota po měření první metodou je o 20 cm vyšší než s použitím druhé metody. Hodnota tohoto parametru má vliv na dobu trvání skoku, vyšší úchup znamená delší dobu trvání skoku.

Úhel odrazu

Pro správný převod energie rozběhu do tyče je užitečné zkoumat úhel odrazu. Uvádí se jako úhel mezi zemí a směrem, kterým se po odrazu pohybuje těžiště. Větší úhel znamená vyšší odraz, ale také větší ztrátu rychlosti při odrazu. Optimální hodnota úhlu odrazu různých atletů se liší například v závislosti na jejich tělesné výšce.

Úhly v kloubech při odrazu

Pozice těla při odrazu je pro přenos energie do tyče také důležitá. Pokud je atletův trup nakloněn dopředu, jeho ramena a paže jsou v lepší pozici pro roztlačení tyče. Naopak pokud je atlet v záklonu, jeho boky po odrazu snáze ujedou směrem dopředu a ramena zůstanou vzadu. To má za důsledek horší roztlačení tyče.

Doba trvání skoku

Doba trvání skoku je vymezena odrazem od země a opuštěním tyče. Na hodnotu tohoto parametru má vliv mnoho faktorů. Technika je jedním z nich, standardně se atleti odráží od tyče vzpaženou paží, jsou ovšem atleti, kteří se tyče pustí dříve. Doba skoku za použití klasického odražení se vzpaženou paží se pohybuje okolo 1,3 s, nestandardní technika může tuto hodnotu stlačit až k 0,7 s. Tyto hodnoty se týkají mužského finále z Mistrovství světa v atletice 2017. Dále má na délku skoku vliv tvrdost tyče, tvrdší tyč se snáze - a tedy i rychleji - narovná a skok tak trvá kratší dobu.

Převýšení

S opuštěním tyče souvisí i převýšení. Jedná se o rozdíl zdolané výšky a výšky úchopu nad zemí. Atleti světové úrovně převýšují okolo 130 cm, ženy okolo 70 cm. Převýšení je jedním z hlavních parametrů, které určují výkonnost atleta. Na jeho hodnotu má vliv jak fyzická zdatnost, tak technické provedení skoku.

1.2 Zpracování obrazu

1.2.1 Detekce člověka

1.2.2 Tracker

1.2.3 Detekce kostry

2. Návrh řešení

Program lze spustit ve dvou módech - analýza videa a analýza modelu.

Při analýze videa dostane program na vstupu video, ve videu najde atleta a s využitím jeho pozice detekuje jeho kostru v jednotlivých snímcích. Z detekované kostry vytvoří 3D model atletova těla, který reprezentuje atletův pohyb.

Analýza pohybu modelu probíhá stejně při zpracování videa i samotného modelu. Podle pohybu jednotlivých částí těla atleta se vypočítají hodnoty biomechanických parametrů, které se uloží a zobrazí uživateli.

2.1 Souřadné systémy

Pro reprezentaci pozic objektů je potřeba zvolit vhodné souřadné systémy, aby nezkreslovaly hodnoty parametrů.

2.1.1 Snímek

Souřadný systém ve snímku má počátek v levém horním rohu. Bod (x,y) je ve vzdálenosti x pixelů od levé strany snímku a y pixelů od horní strany snímku.

2.1.2 Video

Kamera typicky není při natáčení skoků statická, tudíž je potřeba reprezentovat pohyb objektů jinak než pouze jejich pozici ve snímcích videa.

Rozhodl jsem se pro určení pohybu podle vzájemné pozice objektu a pozadí videa v souřadném systému snímku. Polohu pozadí považuji za statickou. Jedná se o jistou formu projekce 3D prostoru do 2D, která zkresluje realitu, tento systém lze reprezentovat jako souřadnice v panoramatické fotce.

Pozice objektu v prvním snímknu, v němž ho detekuju, je počátkem souřadného systému. Nejprve určím pozici levého horního rohu snímknu, v němž se nachází objekt, jehož pozici chci určit. Tuto pozici určím vzhledem k pozici pozadí. Společně s pozicí objektu v tomto snímknu získám pozici objektu vůči jeho pozici ve snímknu, v němž jsem ho detekoval poprvé.

2.1.3 Prostor

Model atleta umístím do 3D prostoru, který se podobá reálnému světu, ale využívá souřadný systém videa, tudíž realitu do jisté míry zkresluje.

Jednotkou souřadného systému 3D modelu je pixel. Odhad reálných vzdáleností z videa je poměrně komplikovaný a pro základní parametry skoku bude tento systém dostačující. Pro případnou konverzi na metry lze systém vhodně přeskálovat v budoucnu. Musí se ovšem brát v potaz vzdálenost atleta od kamery, jelikož se jeho velikost v průběhu videa mění. Pro přesnější konverzi bych tedy musel zvolit jiný způsob určení pohybu objektů ve videu.

Kloub výsledného 3D modelu reprezentují jako uspořádanou trojici (x,y,z) , počátek této soustavy umístím do místa kotníku odrazové nohy při odrazu atleta. Ideální by bylo počátek umístit na zadní hranu kastlíku. Kastlík je schovaný za

doskočištěm, tudíž bych jeho pozici musel odhadovat na základě detekce tyče, což by mohlo být nepřesné.

Hodnota první složky určuje horizontální vzdálenost bodu od počátku ve směru rovnoběžným s osou rozběžiště. Tato hodnota stoupá ve směru rozběhu.

Hodnota druhé složky určuje horizontální vzdálenost bodu od počátku ve směru kolmým na osu rozběžiště. Tato hodnota stoupá ve směru doprava z pohledu atleta při rozběhu.

Hodnota třetí složky určuje vertikální vzdálenost bodu od počátku. Tato hodnota stoupá ve směru vzhůru.

2.2 Vstup programu

Vstupem programu je video skoku o tyči pořízené běžnými prostředky, jakými jsou mobilní telefon nebo tablet.

Videa skoku o tyči se nejčastěji natáčí ze strany. Kamera se standardně nachází na kolmici k rozběžišti, která prochází místem odrazu. Můj program očekává na vstupu video ze strany, pro přesnost hodnot výsledných parametrů je vhodné, aby byla kamera na úrovni odrazu. Pro potřebu přesnější analýzy běhu atleta je lepší umístit kameru na úroveň poloviny rozběhu. Kamera není statická, většinou se pohybuje s tělem atleta.

Při trénincích se kamera nachází ve vzdálenosti okolo 10 metrů od rozběžiště, závodní skoky jsou ve většině případů natáčeny z tribuny, tudíž je vzdálenost větší (v rádu desítek metrů). Velikost atleta ve videu nemá na fukncionalitu programu vliv, ale detekce menší postavy může vyústit v horší přesnost. [Otestovat program na videu, kam se atlet nevejde]

2.3 Analýza videa

Vstupem programu je video skoku o tyči. Toto video program nejprve zpracuje do vhodné reprezentace, následně ve videu nalezne atleta. Na základě pozice atleta v jednotlivých snímcích detekuje kostru atleta, kterou převede do 3D modelu. Podle pohybu atletova těla, který je reprezentovaný získaným modelem, analyzuje parametry skoku a určí jejich hodnoty. Výstupem programu je model atletova těla a hodnoty definovaných parametrů. Tento výstup následně uloží a zobrazí uživateli.

Program lze spustit na počítači s Unixovým operačním systémem.

2.3.1 Zpracování videa

Prvním krokem k provedení analýzy je zpracování videa. Aby video nebylo otevřené po celou dobu analýzy, rozhodl jsem se při analýze pracovat s jednotlivými snímky, které si ukládám při jediném průchodu videem. Rozlišení videa může být libovolné, ale z důvodu časové náročnosti rozlišení snímků videa před analýzou zmenším.

2.3.2 Nalezení atleta

Pro nalezení atleta stačí určit snímek, ve kterém je atlet dobře vidět a jeho pozici v něm. Pozici atleta bude reprezentovat jeho ohraničující rámeček, který je vyznačený na obrázku 2.1.



Obrázek 2.1: Ohraničující rámeček.

Obrázek obsahuje ohraničující rámeček atletova těla ve snímku videa [data/1.MOV](#)

Pro nalezení atleta jsem zvolil dva přístupy.

Prvním z nich je manuální. Uživatel po spuštění analýzy vybere snímek, ve kterém je atlet dobře vidět. Následně vyznačí do videa ohraničující rámeček, do něhož se vejde postava atleta.

Druhým přístupem je automatické nalezení atleta. Při automatické detekci atleta je potřeba detektovat postavy ve snímku videa. Výsledkem této detekce je množina ohraničujících rámečků reprezentujících postavy ve snímku. Z těchto postav je potřeba vybrat tu, která reprezentuje atleta.

Atlet provádí na videu specifický pohyb. Nejprve se rozeběhne a poté provede skok, tedy ohraničující rámeček postavy atleta by se ve videu měl pohybovat horizontálně bez výrazných výkyvů ve vertikálním směru, následně se pohybovat směrem vzhůru a nakonec směrem dolů.

Pomocí trasování objektů mohu ve videu zkoumat pohyb postav a na základě jejich reálného pohybu filtrovat postavy, které mohou reprezentovat atleta.

V prvním snímku detekuju postavy a uložím si je do seznamu, následně budu zkoumat jejich pohyb. Jejich pohyb budu zkoumat pomocí trasování objektů v souřadném systému videa.

Pokud se postava v horizontálním směru nepohně za určitou dobu alespoň o danou vzdálenost, smažu ji ze seznamu, jelikož nereprezentuje atleta. Jakmile je seznam postav prázdný, spustím detekci postav znova v právě zpracovávaném snímku.

V opačném případě považuji postavu za atleta. Tímto okamžikem hledání atleta ve videu končí, jelikož vím, ve kterém snímku jsem atleta detekoval poprvé a jaký byl jeho ohraničující rámeček v daném snímku.

Trasování postav

Trasování postav ve videu používám k odlišení atleta od ostatních postav. Trasování atleta provádím také při detekci kloubů kostry, ale modifikovaným způsobem, který je popsáný v sekci 2.3.3.

Pro trasování postav ve videu stačí knihovní implementace trackerů, velikost postav se ve videu dramaticky nemění a jejich pohyby jsou plynulé.

2.3.3 Detekce kostry

Trasování atleta

Trasování atleta při rozběhu nedělá knihovním trackerům problém. Nepřesné trasování nastává při skoku. Důvodem těchto nepřesností je rotace atletova těla. Tato rotace je při skoku výrazná a při vodorovné pozici trupu nemusí atletovo tělo zabírat věšinu ohraničujícího rámečku, což způsobí, že tracker začne trasovat pozadí videa a ztratí atleta. Tato vlastnost a její napravení je vidět na obrázku 2.2.

Modifikací vedoucích ke zlepšení přesnosti jsem zkusil několik.

Nejprve jsem otáčel snímky videa podle naklonění trupu v předešlém snímku. K určení tohoto naklonění jsem použil detekovanou kostru atletova těla.

Následně jsem aktualizoval tracker, aby trasoval pouze trup. Jeho pozici jsem opět získával z detekované kostry. Tyto metody selhávaly při chybné detekci kostry, ačkoliv jsem se snažil implementovat kontrolní mechanismus. Například jsem neaktualizoval tracker na kostře, jejíž trup byl příliš daleko od právě trasovaného trupu. [OBRÁZEK TOHOTO PROBLÉMU]

Nakonec jsem implementoval metodu, která nezávisí na detekci kostry atleta.

Atletovo tělo netrasuji jako celek, ale po částech. Původní ohraničující rámeček rozdělím na mřížku menších rámečků. Každý rámeček trasuji zvlášť a průběžně aktualizuji ty, které atleta ztratí.

Po každých několika snímcích kontroluji pohyb jednotlivých rámečků ve videu od předešlé kontroly. Určím rámeček, který se pohnul nejvíce. Pohyb rámečku musí být horizontálně ve stejném směru, jako je rozběh atleta. Následně posunu původní mřížku tak, aby odpovídala pozici nejvíce se pohybujícího rámečku. [OBRÁZEK s podrobnějším popisem] Do mřížky poté přesunu rámečky, jejichž trasování selhalo, na své původní místo. [OBRÁZEK před a po]

Detekce kloubů těla

Pro určení atletovy kostry stačí detektovat klouby, které následně spojím úsečkami. Klouby detekuju pomocí konvoluční sítě. Jelikož má implementace očekává, že na snímku je jediná postava, bylo pro dostatečnou kvalitu detekce kloubů potřeba vyříznout ze snímku okno, v němž se nachází atlet a příslušně okno rotovat, aby byl atlet v co nejvzpřímenější pozici. Sít, kterou používám, je totiž natrénovaná na databázi MPII Human Pose dataset (Andriluka a kol., 2014), v níž je většina postav ve vzpřímené poloze.

Určení pozice atletova okna provádím s použitím rámečků na těle atleta, kterého trasuji. Rámečky uzavřu do co nejmenšího rámečku a v něm naleznu střed. To je střed prvního okna pro detekci kostry. Střed okna v následujících snímcích posouvám směrem ke kyčlům atleta. Tento posun získám z pozice rámečku



Obrázek 2.2: Ztráta atleta při použití různých trackerů.

Obrázek znázorňuje použití jednoho trackeru (vlevo) a mřížky trackerů, která je implementovaná v programu (vlevo). Použité snímky jsou vybrané z videa [data/33.MOV](#).

obsahujícího trackery atletova těla a pozice kyčlí kostry detekované v předchozím snímku. [OBRÁZEK posunu okna před a po aktualizaci mřížky] Aby se okno neposunulo příliš daleko od těla atleta při chybné detekci kostry, posouvám střed okna jen uvnitř rámečku obsahujícího trackery atletova těla. [OBRÁZEK detekce chybné postavy a následné napravení]

Rotaci okna provádím podle náklonu trupu poslední validně detekované kostry. Validně detekovanou kostrou rozumím kostru, které jsem detekoval všechny klouby. Abych zamezil rotacím podle chybné detekce, aktualizuju poslední známý úhel náklonu trupu jen v případě, že se od předchozího příliš neliší. Pokud tedy při skoku detekuju kostru postavy, která stojí u místa odrazu, neaktualizuju úhel rotace a okno rotuji podle posledního známého naklonění trupu atleta. [OBRÁZEK]

Pokud se mi nepodaří detekovat celou kostru, zkusím okno rotovat o určitý úhel oběma směry a vyberu ten s nejúspěšnější detekcí, co se počtu detekovaných kloubů kostry týče.

Jelikož atlet mění v průběhu videa velikost - standardně se zvětšuje, protože se přibližuje ke kamere - je potřeba okno postupně zvětšovat. První okno je dané původním ohraňujícím rámečkem, který pro zvětšení konstantou. Velikosti následujících oken určuji podle dosud největší validně detekované kostry. Velikost se rovná vzdálenosti dvou nejvzdálenějších kloubů kostry. Okno pro detekci kostry je čtvercové, jehož strana má délku velikosti dosud největší validně detekované kostry, jejíž velikost zvětší konstantou. Okno pro první detekci nemusí být čtvercové, odvíjí se od výsledku nalezení atleta ve videu.

2.3.4 Konverze kostry do 3D modelu

Konverzi detekované kostry do 3D modelu provedu pomocí algoritmu pro převod mezi souřadnými systémy videa a 3D modelu. Tyto souřadné systémy jsou popsány v sekci 2.1 a implementace algoritmu v sekci 3.2.

2.3.5 Analýza parametrů

Vzhledem k vlastnostem mnou zvoleného modelu nebude vhodné analyzovat některé parametry skoku. Rozeberu proveditelnost analýzy biomechanických parametrů popsaných v sekci 1.1.2.

Délka jednotlivých kroků rozběhu

Délka jednotlivých kroků rozběhu je velice citlivá na úhel, pod kterým atleta snímám a jeho vzdálenost od kamery. Vliv vzdálenosti by bylo možné vyřešit reprezentací délky kroku relativně vůči výšce postavy. Ovšem úhel osy kamery vůči rozběžnosti v programu nijak neřeším, tudíž se této nepřesnosti nelze jednoduše zbavit.

Proto jsem se rozhodl tento parametr neanalyzovat.

Místo délky kroku jsem se rozhodl implementovat zjištování hodnoty doby trvání jednotlivých kroků. K zisku této hodnoty stačí najít snímky, ve kterých se nižší noha začíná pohybovat směrem vzhůru. Tím získám (po jednoduché filtrace) momenty dokončení odrazů jednotlivých kroků. Doba, která uplyne mezi následujícími kroky je výsledná hodnota.

Doba oporové fáze kroků

Dobu oporové fáze kroku lze analyzovat mírným rozšířením analýzy doby trvání jednotlivých kroků. Při spuštění stejného algoritmu od konce videa získám momenty došlapů. Spolu s momenty odrazů lze určit dobu oporové fáze jednotlivých kroků.

Vzhledem k tomu, že jeho přesnost závisí na frekvenci snímků videa jsem se analýzu tohoto parametru rozhodl neanalyzovat.

Náběhová rychlosť

Náběhovou rychlosť nelze analyzovat v průběhu celého rozběhu příliš přesně z důvodu měnícího se úhlu, pod kterým kamera snímá atletův rozběh. V momentu odrazu lze rychlosť reprezentovat poměrně přesně, ale jen v jednotkách závislých na pixelech. Pro převod na jednotky s větší vypovídající hodnotou by bylo možné využít délku tyče nebo výšku atleta. Pro implementaci této metody by bylo potřeba, aby uživatel zadal příslušný parametr - výšku atleta nebo délku tyče - a výsledná přesnost by nebyla příliš dobrá, jelikož se náběhová rychlosť u různých skoků liší nepatrně. Lepším způsobem pro analýzu tohoto parametru tak zůstává využití fotobuňek nebo radaru.

Místo analýzy tohoto parametru jsem zkoumal ztrátu horizontální rychlosti ramen a boků při odrazu. Tato změna nezávisí na použitých jednotkách, takže není potřeba odhadovat vzdálenost. Pro zisk těchto hodnot porovnávám změnu pozice ramen a boků daný časový úsek před odrazem, při odrazu a stejný časový úsek po něm.

Výška boků v průběhu rozběhu

Výška boků bez jakéhokoliv škálování v závislosti na velikosti atleta ve videu není příliš vypovídající pro porovnání začátku a konce rozběhu. Vliv na výsledný skok mají spíše lokální výkyvy, především ty, které nastávají v konci rozběhu. Tím pádem je hodnota tohoto parametru užitečná i bez škálování naměřených hodnot.

Výšku boků jsem tedy mezi implementované parametry zahrnul, pro zisk jeho hodnoty průměruji výšku levé a pravé kyčle.

Místo odrazu

Přesnost tohoto parametru závisí také na zvoleném modelu, kterým reprezentuji tělo atleta, tedy kolik bodů na těle detekuji. Nejčastěji se udává jako vzdálenost špičky chodidla od zadní hrany kastlíku. Pozici špičky chodidla je složité odhadnout, pokud znám jen pozici kotníku. Přesnost pdhadu hrany kastlíku by na hodnotu tohoto parametru měla také značný vliv.

Pozice odrazu se jednoduše a přesně určí pouhým okem, především pokud jsou poblíž místa odrazu na zemi značky, proto jsem tento parametr neimplementoval.

Výška úchopu

Podobně jako v předchozím případě lze hodnotu tohoto parametru poměrně přesně určit pouhým okem při znalosti délky tyče. Proto nebylo potřeba analýzu tohoto parametru implementovat.

Úhel odrazu

Jelikož je osa kamery při odrazu většinou kolmá na směr rozběhu, je možnost získání přesné hodnoty tohoto parametru solidní. Úhel určím podobně jako ztrátu rychlosti. Porovnám pozici boků daný časový úsek před odrazem, při odrazu a stejný časový úsek po něm. Na základě rozdílů těchto pozic určím úhel rozběhu (jeho konce) a skoku (jeho počátku) vůči horizontální ose. Po odečtení úhlu rozběhu od úhlu skoku získám úhel odrazu.

Úhel odrazu jsem se tedy rozhodl analyzovat.

Úhly v kloubech při odrazu

Úhly v kloubech kostry lze z modelu získat snadno, ale lze je odhadnout poměrně přesně i bez analýzy programem - alespoň pro účely porovnání skoků. Nejedná se o tak důležitý parametr jako úhel odrazu, tedy implementaci analýzy úhlů kloubů těla ponechám jako možné rozšíření programu do budoucna.

Doba trvání skoku

Pro zisk hodnoty tohoto parametru by bylo užitečné detekovat tyč. Tuto funkcionality jsem neimplementoval, tudíž tento parametr není mezi analyzovanými.

Převýšení

Pro určení míry převýšení je také vhodné detekovat tyč. Druhou možností je převod souřadného systému na metry a zadání výšky úchopu. Poté by bylo možné převýšení určit. V budoucích verzích programu by se tento parametr mohl objevit. Zatím jsem ho neimplementoval.

Další implementované parametry

Nad rámec popsaných parametrů jsem se rozhodl implementovat analýzu následujících parametrů.

Úhel došlapu jednotlivých kroků Parametrem souvisejícím s dobou oporové fáze kroku je úhel došlapu. Jedná se o úhel mezi kotníkem, kyčlí a vertikálou. Aproximuje vzdálenost došlapu před těžiště atleta. Pro zisk hodnoty tohoto parametru použiji metodu získávání snímků, ve kterých dochází k došlapu.

Náklon trupu Jedná se o náklon trupu v rovině určené vertikálou a osou rozběžiště. Hodnota tohoto parametru není příliš přesná na začátku rozběhu, ale s postupem času se přesnost zvětšuje díky lepšímu úhlu záběru těla atleta.

Důležité momenty skoku

Kromě parametrů ve videu detekuji podstatné momenty skoku, mezi něž považuji začátek rozběhu, odraz a moment kulminace boků nad laťkou.

2.4 Analýza modelu

Aby uživatel nemusel video analyzovat znovu, je mu umožněno načtení modelu ze souboru, který je popsaný v sekci 2.5.

Funkcionalita programu je velice podobná, narozdíl od analýzy videa se na začátku načte také soubor s uloženým modelem. Neprovádí se hledání atleta ve videu, detekce kostry, ani konverze kostry do 3D modelu. Analýza parametrů probíhá stejně jako při analýze videa, model se neukládá, ukládají se jen parametry, aby se jejich hodnoty daly využít pro případnou analýzu.

Kostru z 3D modelu získá program s využitím konverzí souřadních systémů, kterou popisují v sekci 3.2.

2.5 Výstup programu

Výstupem programu je model reprezentující pohyb atleta v průběhu rozběhu i skoku a hodnoty definovaných parametrů, které jsou

- doba trvání jednotlivých kroků,
- ztráta rychlosti ramen a boků,
- výška boků,
- úhel odrazu,
- úhel došlapu jednotlivých kroků,
- náklon trupu a
- důležité momenty skoku.

2.5.1 Uložení výstupu

Parametry Hodnoty parametrů vypíšu do souboru pro případnou hlubší biomechanickou analýzu. Jedná se o .csv soubor, jehož formát je následující.

Parametry ukládám po sloupcích. První sloupec obsahuje relativní čas vůči momentu odrazu v každém snímku videa, následující sloupce obsahují parametry.

První řádek souboru obsahuje název analyzovaného videa, druhý názvy parametrů, počínaje od druhého sloupce. Ve třetím řádku jsou vypsané jednotky, ve kterých jsou hodnoty parametrů určeny.

Parametry, které reprezentuje jedna hodnota, zabírají jen první řádek hodnot.

Parametry, které neodpovídají snímkům, ale obsahují více hodnot (například doba trvání jednotlivých kroků), jsou uloženy postupně od prvního řádku, dokud jejich hodnoty znám.

Parametry, pro které znám hodnotu v každém snímku, jsou uloženy v řádcích odpovídajících příslušným snímkům videa. [UKÁZKA]

Model Po analýze videa se výsledný model uloží do textového souboru. První řádek obsahuje název analyzovaného videa.

Následně se v souboru opakují části reprezentující detekce jednotlivých snímků. Na prvním řádku této části je číslo snímku videa, následuje pozice levého hodního rohu snímku v souřadnicích videa převedených do 3D - hodnota x -ové osy roste směrem doprava. Na dalších řádcích jsou vypsány klouby kostry těla v souřadnicích snímku převedených do 3D. Tyto klouby jsou uloženy postupně, podle zvoleného modelu reprezentace těla, ten je popsán v sekci 3.1. [UKÁZKA]

Po analýze modelu se model do souboru neukládá.

2.5.2 Zobrazení výstupu uživateli

Výstup programu je vhodné uživateli přehledně zobrazit. Rozhodl jsem se pro vlastní prohlížeč snímků videa. Mezi jednotlivými snímky se lze pohybovat dopředu i dozadu. Ve výchozím režimu se zobrazí video se zakreslenou kostrou, jejíž zakreslení do snímku lze vypnout a znova zapnout.

Při prohlížení daného snímku vypíše program do konzole hodnoty příslušných parametrů. Parametry jsem rozdělil do kategorií podle části skoku, při které jsou podstatné. Například úhel došlapu je irrelevantní při skoku, ale je důležitý při rozběhu. Tedy při snímku rozběhu se vypíše například úhel naklonění trupu v daném snímku a doba trvání právě probíhajícího kroku. Parametry související s odrazem vypíše program při zobrazení snímku, který reprezentuje moment odrazu. [OBRÁZKY]

Po ukončení prohlížeče snímků zobrazím uživateli hodnoty parametrů zanesených do grafů, které se načtou z vygenerovaného souboru.

2.6 Programové vybavení

2.6.1 Počítačové vidění a zpracování obrazu

Pro potřeby počítačového vidění a zpracování obrazu lze využít několik knihoven.

OpenCV OpenCV (Bradski, 2000) je jednou z nejrozšířenějších open-source knihoven. Mezi její výhody patří všeobecnost. Je vhodnou volbou pro zpracování obrazu, videí a vytváření modelů počítačového vidění a strojového učení. OpenCV vznikla již v roce 2000, což je jedním z důvodů, proč ji využívá a udržuje mnoho vývojářů. Na internetu lze tedy najít mnoho informací o naprosté většině funkcionality, kterou tato knihovna disponuje.

OpenCV lze spustit na různých platformách a je vhodnou volbou i pro mobilní zařízení, což může být užitečné pro budoucí rozšíření aplikace. Tuto knihovnu lze použít v programu psaném v jazycích C, C++, Python nebo Octave.

OpenVINO OpenVINO je knihovna z dílny společnosti Intel. Existuje ve dvou verzích, jako open-source a jako distribuce spravovaná právě společností Intel. Knihovna disponuje solidní zásobou modelů hlubokého učení pro potřeby počítačového vidění a jejich optimalizací pro procesory společnosti Intel.

Ačkoliv se vývojáři snaží podporovat i jiné procesory, může být spuštění této knihovny na procesorech s jinou architekturou probematické. Možnosti knihovny pro zpracování obrazu jsou omezené, takže se často používá v kombinaci s OpenCV.

API této knihovny je určené pro jazyky C, C++ a Python.

VisionWorks Společnost NVidia vyvinula knihovnu VisionWorks, která využívá rychlosti grafických karet. Tato knihovna slouží především k detekci a trasování objektů. Má výborné výsledky v oblasti autonomního řízení, což není příliš použitelné v mému programu.

Vision Workbench NASA spravuje vlastní knihovnu Vision Workbench zabývající se počítačovým viděním a zpracováním obrazu. Hlavním cílem této knihovny je zpracování obrazu a počítačové vidění pro vesmírné roboty.

2.6.2 Grafická reprezentace výstupu

Užitečným prvkem při analýze videozáznamu je přehrávač videa. Pro mé potřeby by bylo vhodné, aby se pro daný snímek zobrazovaly hodnoty parametrů, takže přehrávač videa nahradím prohlížečem snímků videa.

Pro analýzu parametrů a jejich případné porovnání je vhodné jejich hodnoty zanést do grafů.

gnuplot Jedním z nejrozšířenějších programů pro tvorbu grafů je gnuplot (Williams a kol., 2013). Program se spouští z příkazové řádky a běží na několika platformách. Jedná se o program, jehož první verze vznikla už v roce 1986, takže má, podobně jako OpenCV, velké množství uživatelů, kteří s ním mají bohaté zkušenosti.

Gnuplot zpracovává vstup z příkazové řádky nebo skriptů, které popisují, jaká data se mají vykreslit. Příkazy nejsou psané v žádném programovacím jazyce, ale mají vlastní syntaxi.

MATLAB Velké množství vědců používá pro práci s daty MATLAB (MATLAB, 2010). Jedná se o programovací jazyk, který se specializuje například na numerické výpočty, práci s maticemi nebo zanášení dat do grafů.

MATLAB je možné provázat s jinými programovacími jazyky, mezi něž patří C, Java nebo Python.

Matplotlib Stále větší oblibě se těší programovací jazyk Python, který disponuje knihovnou pro generování grafů. Tato knihovna se jmenuje Matplotlib (Hunter, 2007), která se používá pouze pomocí jazyka Python. Standardním způsobem práce s touto knihovnou je přes modul Pyplot, jehož funkcionalita je podobná MATLABu. Pyplot disponuje množinou funkcí, které umí vykreslit různé typy grafů. Výhodou použití Pythonu je také snadné spouštění skriptů, které není potřeba manuálně kompilovat před jejich spuštěním.

Skripty psané v Pythonu lze spouštět přímo z kódu jazyka C++, což umožňuje automatické zobrazení grafů po doběhnutí analýzy videa skoku o tyči.

3. Realizace

3.1 Zvolené prostředky

Pro potřeby analýzy videa jsem použil jazyk C++ a knihovnu OpenCV. Jelikož je analýza výpočetně náročná, dal jsem z rychlostních důvodů přednost jazyku C++ před jazykem Python. Také mám s jazykem C++ více zkušeností.

Tracker, který jsem zvolil pro trasování objektů je `cv::TrackerCSRT`, jeho implementace je založena na využití Discriminative Correlation Filter with Channel and Spatial Reliability (Lukežič a kol., 2018).

Pro zobrazení parametrů v grafické podobě jsem se rozhodl využít Matplotlib. Pro využití této knihovny není potřeba znát speciální syntaxi, základní znalosti Pythonu jsou dostačující. Jelikož je knihovna Matplotlib určena pro Python, je snadné napsat skript, který nejen vygeneruje grafy, ale také zpracuje data, která se mají zobrazit.

Program lze spustit na počítači s Unixovým operačním systémem.

3.2 Konverze souřadných systémů

Souřadné systémy snímku a videa jsou jen posunuté o vzájemnou pozici levého horního rohu snímku a pozadí v prvním snímku. Tudíž jejich konverzi reprezentuje operace posunutí.

Konverze mezi souřadným systémem videa a prostoru je složitější. Jednotka systémů je stejná, tudíž škálování není třeba provádět. První složky bodů v obou systémech spolu korespondují. Druhá složka bodu v souřadném systému videa odpovídá třetí složce bodu v prostoru. Tuto konverzi používám jen pro určení pozice kloubů těla atleta.

Je třeba zohlednit směr rozběhu atleta. Pokud běží doleva, musím převrátit hodnotu první složky. Pokud běží doprava, hodnotu první složky nechám stejnou.

Hodnotu druhé složky nastavím na 0. Odhad pozice kloubu v tomto směru je problematický, jelikož se atlet při skoku otáčí podél své osy, tudíž její implementaci ponechám do budoucích verzí programu. Pro její správné určení bude také potřeba zohlednit stranu, ze které je video natočeno.

Hodnotu třetí složky v prostoru získám převrácením druhé složky bodu v souřadném systému videa.

Výsledkem tohoto postupu je posunutý prostor, musím ho tedy posunout s využitím pozice kotníku odrazové nohy v momentu odrazu.

Konverzi bodu kostry (x', y') v souřadném systému videa do prostoru lze reprezentovat takto:

$$\begin{aligned} \text{Při běhu doprava: } & (x, y, z) = (x', 0, -y') - d. \\ \text{Při běhu doleva: } & (x, y, z) = (-x', 0, -y') - d. \end{aligned}$$

d značí pozici kotníku odrazové nohy při odrazu v prostoru.

Pozici kotníku odrazové nohy získám až po analýze parametrů, tudíž při analýze pracuji s posunutým prostorem. Na vliv většiny parametrů to vliv nemá, některé poté musím spočítat znova (například výšku boků).

3.3 Struktura programu

Program se skládá z několika tříd, UML diagram popisující strukturu je na obrázku 3.1. Kompletní funkcionalita je přístupná skrz třídu `video_processor` a její metody `process_video` a `process_model`. Tyto metody komunikují se zbylými částmi programu.

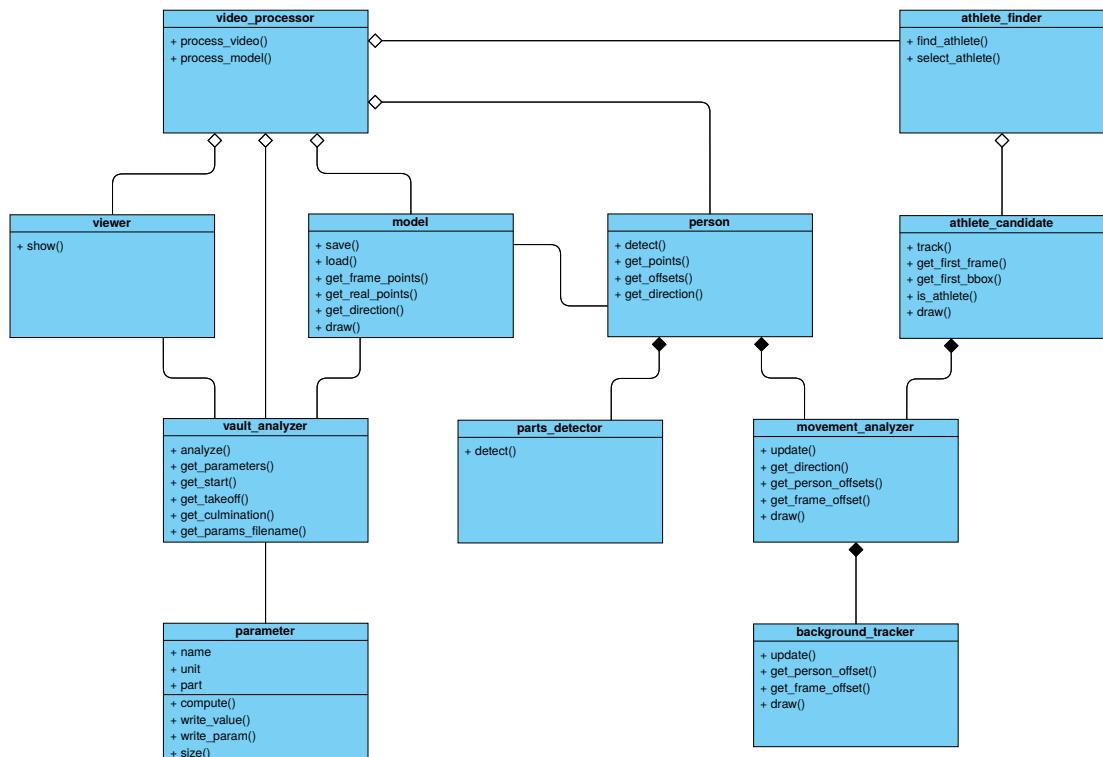
O nalezení atleta se stará třída `athlete_finder`, ta po nalezení atleta vrátí jeho pozici ve videu. K určení atletovy pozice využívá instance třídy `athlete_candidate`, které reprezentují postavy ve videu. Analýzu pohybu jednotlivých postav provádí jednotlivé instance třídy `movement_analyzer` a část zabývající se detekcí pozadí provádí pro každou postavu instance třídy `background_tracker`.

Funkcionalitu týkající se detekcí kostry atleta zajišťuje třída `person`. Pohyb atleta zkoumá instance třídy `movement_analyzer` ve spolupráci s instancí třídy `background_tracker`. O detekci jednotlivých částí těla se stará instance třídy `parts_detector`.

Po detekování kostry těla kovertuje detekce do modelu atletova těla třída `model`, výsledek této detekce využívá pro určení parametrů instance třídy `vault_analyzer`, která určí hodnoty biomechanických parametrů. Parametry reprezentují specializované struktury, které dědí vlastnosti struktury `parameter`. Struktura zobrazující hierarchii parametrů je na obrázku ???. `vault_analyzer` také zajišťuje ukládání parametrů do souboru.

Zobrazování výstupu má za úkol třída `viewer`.

Konstanty, definice vlastních typů a pomocné funkce jsou uloženy v souborech `forward.cpp` a `forward.hpp`.



Obrázek 3.1: UML diagram zobrazující strukturu programu.

3.4 Vstup programu

Programu jsou při jeho spuštění předány parametry. Parametry specifikují mód, ve kterém si uživatel přeje program spustit a obsahují cesty k souborům, které má program zpracovat.

O zpracování parametrů se stará soubor `main.cpp`, který definuje vstupní bod aplikace a volá metody třídy `video_processor`, která zpracuje video, případně model, a výsledek předá k analýze.

3.5 Analýza videa

3.5.1 Zpracování videa

Zpracování videa má na starosti třída `video_processor`.

Video načtu s použitím třídy `cv::VideoCapture`. Z videa získám hodnotu frekvence snímků a jednotlivé snímky videa uložím při průchodu do vektoru 3-rozměrných matic `cv::Mat`, které reprezentují snímky videa.

Z důvodu časové náročnosti videí s velkým rozlišením zmenším každý snímek metodou `cv::resize`. Výsledný snímek videa má rozlišení 720p. Při zpracování videa na šířku je výška videa 720 pixelů, šířka se změní, aby byl poměr stran snímku zachován. Při zpracování videa na výšku je šířka výsledného snímku 720 pixelů, výška se změní v odpovídajícím poměru.

Vzorce pro změnu velikosti snímku videa s rozlišením $x' \times y'$ na výsledné rozlišení $x \times y$, reprezentující 720p jsou následující:

$$\begin{aligned} \text{Na výšku: } & (x, y) = (720 \cdot x' / y', \quad 720) \\ \text{Na šířku: } & (x, y) = (720, \quad 720 \cdot y' / x') \end{aligned}$$

První složka udává šířku snímku, druhá výšku.

3.5.2 Nalezení atleta

Nalezení atleta jsem implementoval dvěma způsoby - manuálním a automatickým. O nalezení atleta se stará třída `athlete_finder`.

Manuální způsob

Tímto způsobem určí uživatel pozici atleta ve snímku. Vybere snímek, ve kterém je atlet v záběru a následně ho uzavře do ohraničujícího rámečku. Tuto funkcionality v grafickém rozhraní nabízí metoda `cv::selectROI`.

Automatický způsob

Abych nalezl atleta automaticky, procházím snímky videa postupně, udržuji si seznam postav ve videu a atleta ve videu. Postavy reprezentuje `std::list`, jehož prvky jsou instance třídy `athlete_finder::athlete_candidate`. Pro `std::list` jsem se rozhodl, jelikož se počet postav ve videu často mění a při této operaci není potřeba `std::list` realokovat. Atleta ve videu reprezentuje instance `std::optional<person>`, její hodnota není validní, dokud nenaleznu atleta.

Dokud nenaleznu atleta, opakuji při zpracování každého snímku následující postup.

Vyfiltruji seznam postav ve videu, abych v seznamu nechal jen postavy, které mohou reprezentovat atleta. Mezi těmito postavami se pokusím najít atleta. Pokud ho najdu, vytvořím instanci třídy `person` na základě snímku, ve kterém jsem ho poprvé detekoval, a jeho pozice v něm. Pokud je seznam postav prázdný a zatím se mi nepodařilo najít atleta, spustím detekci postav v právě zpracovávaném snímku a detekované postavy uložím do seznamu postav.

Postavy, jejichž trasování neselhalo, jsou celé uvnitř snímku [MOŽNÁ ÚPRAVA] a pohybují se, mohou reprezentovat atleta, proto je ze seznamu postav nesmažu. Pokud neuplynulo 0.3 s od první detekce postavy, považuji ji za pohybující se. Po uplynutí této doby se musela od momentu první detekce pohnout v horizontálním směru alespoň o šířku jejího ohraničujícího rámečku, abych ji považoval za pohybující se a nechal ji mezi kandidáty na atleta. Pozici postavy určuje střed jejího ohraničujícího rámečku. Pohyb postav zkoumá třída `movement_analyzer`.

Za atleta zvolím postavu, pro kterou je určen směr pohybu. Směr pohybu určí třída `movement_analyzer` jakmile se postava pohně alespoň o šířku jejího ohraničujícího rámečku v horizontálním směru.

Ohraničující rámečky postav vykreslím do snímku a tyto snímky vrátím pro následné vytvoření videa.

Detekce postav ve snímku Detekci postav ve snímku provádí instance třídy `cv::HOGDescriptor` pomocí knihovního SVM detektoru, který vrátí metoda `cv::HOGDescriptor::getDefaultValueDetector`. Ohraničující rámečky postav ve videu získám metodou `cv::HOGDescriptor::detectMultiScale`. Parametry této metody jsou

- právě zpracovávaný snímek,
- detekované ohraničující rámečky,
- práh detekce,
- posun okna,
- obal okna,
- škálování,
- konečný práh a
- použití seskupovacího algoritmu.

Práh detekce určuje vzdálenost klasifikační roviny SVM detektoru a vlastností právě zpracovávaného okna. Hodnotu tohoto parametru jsem nechal výchozí - tedy 0.

Posun okna určuje velikost kroku mezi okny pro detekci postav ve snímku. Velikost tohoto kroku jsem nastavil na (4,4) pixely.

Obal okna je počet pixelů okolo okna v obou směrech, které detektor zahrnuje do detekce. Velikost obalu jsem zvolil (16,16).

Škálování udává koeficient, kterým se zmenšuje vstupní snímek. Jelikož detekuji různě veliké postavy a okno pro detekci má pevnou velikost, musím měnit velikost snímku a detekci spouštět na různých velikostech. Snímek tedy po každé detekci zmenším daným koeficientem, jehož hodnotu jsem nastavil na 1.05.

Konečný práh není v knihovně dostatečně komentován, proto jsem ponechal výchozí hodnotu 2.

Použití seskupovacího algoritmu udává, zda se před vrácením ohraničujících rámečků postav má použít algoritmus pro jejich seskupení. Tento algoritmus nefungoval příliš dobře, proto seskupení rámečků implementuji sám. Používám modifikaci algoritmu non-maxima suppression (Rosebrock, 2015). Algoritmus jsem implementoval stejně, jen jsem se rozhodl seřadit rámečky podle velikosti, nikoliv podle pozice jejich spodních stran. Důvodem pro tuto změnu byla skutečnost, že metoda detekovala postavu jen na části atletova těla, tudíž pro určení výsledné detekce dávám přednost větším rámečkům. [OBRÁZEK]

Analýza pohybu postav O analýze pohybu postav se stará třída `movement_analyzer`, stejná třída se stará i o pohyb atleta, jehož trasování je modifikované, tudíž je analýza pohybu blíže popsána v sekci 3.5.3.

Při analýze atletova pohybu předávám třídě `movement_analyzer` vektor všech rámečků mřížky. Postavy při hledání atleta mřížku nemají, tudíž předám jedno-prvkový vektor obsahující ohraničující rámeček postavy.

3.5.3 Detekce kostry

Jelikož může být postav ve videu více, bylo potřeba pro přesnost detekce kostry atleta ze snímku vyříznout okno, ve kterém se atlet nachází. Pozici tohoto okna určuji trasováním atleta. Detekci kostry atleta zajišťuje instance třídy `person`, která reprezentuje atleta.

Video se zpracovává postupným procházením snímků videa. V každém snímku určím pozici atletova těla a následně detekuji klouby kostry. Do snímku zakreslím detekce a po průchodu celého videa snímky se zakreslenými detekcemi vrátím.

Pozice detekovaných kloubů atletova těla v jednotlivých snímcích udržuje instance třídy `person`. Posun snímků pro převod těchto pozic do souřadného systému videa obstarává třída `movement_analyzer`.

Trasování atleta

Jak jsem již popisoval v sekci 2.3.3, bylo potřeba trasování atleta modifikovat, aby byla kvalita trasování při skoku dostatečná. Rozhodl jsem se pro rozdelení původního ohraničujícího rámečku atleta na mřížku 3×3 menších rámečků. Chtěl jsem jedním z menších rámečků pokrýt střed ohraničujícího rámečku, proto jsem se rozhodl pro tento počet. Každý rámeček v mřížce trasuji samostatně, trackery jednotlivých rámečků jsou uloženy v 1D vektoru. Kvůli analýze pohybu rámečků při aktualizaci mřížky ukládám pozice rámečků do vektoru, který pro každý snímek obsahuje vektor pozic rámečků v daném snímku.

V prvním snímku, kde jsem detekoval atleta inicializuji tracker pro každý rámeček mřížky.

Ihned po inicializaci a v následujících snímcích určím pozici jednotlivých rámečků. Pokud trasování všech rámečků mřížky selže, ukončím trasování atleta a v následujících snímcích kostru nedetekuji.

Vzhledem k tomu, že atleta trasuji po částech se může stát, že některé trackery začnou trasovat pozadí videa. Abych je přesunul na správné místo, zkoumám pohyb jednotlivých trackerů za poslední 2 snímky.

Vyberu rámeček, jehož pozice se za poslední 2 snímky nejvíce změnila. Kontroluji, aby horizontální směr odpovídal směru rozběhu atleta. K tomuto rámečku přesunu mřížku, aby v ní byl nejvíce se pohybující se rámeček na původním místě. Musím také zajistit, aby byla mřížka uvnitř snímku. Mřížku opět rozdělím na menší rámečky a inicializuji v ní trackery, které atleta ztratily.

Tyto trackery určím tak, že posun jejich rámečků je menší než $1/4$ posunu nejvíce se pohybujícího rámečku. Při skoku atlet švihá nohou směrem dopředu, pohyb této nohy vůči pozadí je přibližně dvojnásobný vůči pohybu trupu, tudíž jsem musel zvolit menší hodnotu než $1/2$, abych zamezil přesunu validních trackerů z trupu na nohu.

Po aktualizaci rámečků je 2 snímky neaktualizuji, aby do jejich pohybu nebyla započítána jejich aktualizace.

Atletova pozice se v průběhu videa mění, snažil jsem se implementovat i změnu velikosti mřížky podle velikosti těla atleta, ale přesnost trasování se zhoršila, proto škálování mřížky neprovádím.

Analýza pohybu atleta

Pro určení pozice atleta v souřadném systému videa využívám třídu `movement_analyzer` a `background_tracker`. Tato třída analyzuje také pohyb postav, každá instance analyzuje pohyb jediné postavy.

Atlet je ve snímku reprezentován několika rámečky, postava jedním, ale ten zabalím do vektoru a následný postup je pro oba případy stejný. Pozicí postavy ve snímku rozumím střed nejmenšího obdélníku, do něhož lze rámečky uzavřít. Rámečky ve snímku, snímek a číslo snímku dostane analyzátor pobytu na vstupu.

V prvním snímku, ve kterém chci určit pozici atleta, inicializuji tracker pozadí. Část pozadí, kterou budu trasovat uzavřu do ohraničujícího rámečku, který má velikost $x/4 \times y/2$, kde x je šířka snímku, y výška. Tento rámeček umístím k levé nebo pravé straně snímku, podle toho na jaké straně je průnik atleta a pozadí menší. Přednost má levá strana. Rámeček umístím tak, aby nad ním i pod ním bylo stejně místa. Tím se vyvaruji trasování jednobarevného nebe, případně tartanu, pokud bych pozadí umístil nahoru, případně dolů.

Při zpracování každého snímku zjistím pozici levého horního rohu snímku v souřadném systému videa, díky čemuž snadno určím pozici libovolného objektu ve snímku.

Pozice pozadí ve videu je $d_0 = b_0 - p_0$, b_0 udává pozici středu pozadí v tomtéž snímku, p_0 je pozice atleta v prvním snímku, ve kterém ho detekuji. Pozice t_0 levého horního rohu prvního snímku je rovna $-p_0 = d_0 - b_0$.

Při analýze dalšího snímku znám pozici atleta p_1 ve snímku. S využitím trackeru určím pozici pozadí b_1 ve snímku a jeho pozici $d_1 = d_0$ ve videu. Ze znalosti pozice pozadí získám pozici levého horního rohu snímku $t_1 = d_1 - b_1$. Takto získám pozici levého horního rohu každého snímku.

Jelikož se záběr kamery pohybuje, je potřeba pozadí aktualizovat, jakmile se dostane mimo záběr. S touto aktualizací musím aktualizovat pozici nového pozadí ve videu. Mám tedy pozadí, které se dostalo mimo záběr. Jeho pozice ve videu je d_i , pozice ve snímku b_i . Inicializuji nové pozadí s pozicí n_i ve snímku. Pozice nového pozadí ve videu se určí jako $d_i + n_i - b_i$. Dále pracuji s touto hodnotou místo d_i a novým pozadím videa. [OBRÁZEK]

Pozadí přesunu také jakmile do oblasti pozadí vběhne atlet. Pokud pozadí bez průniku s atletem nelze vytvořit, nezacyklím se, jen budu v každém snímku vytvářet nové pozadí, aby byl průnik minimální.

Podle pohybu atleta ve videu také určím směr jeho rozběhu pro následný převod kostry do 3D modelu. Směr rozběhu určím, jakmile se pozice atleta změní alespoň o šířku jeho těla od první pozice, kde jsem atleta detekoval.

Detekce kloubů těla

Klouby těla jsou určeny databází MPII Human Pose dataset (Andriluka a kol., 2014). Jedná se o 16 bodů těla, ale posledním je pozadí, které mohu ignorovat. Používám tedy jen prvních 15 bodů těla. Jedná se o

0. hlavu,
1. krk,
2. pravé rameno,
3. pravý loket,
4. pravé zápěstí,
5. levé rameno,
6. levý loket,
7. levé zápěstí,
8. pravou kyčel,
9. pravé koleno,
10. pravý kostník,
11. levou kyčel,
12. levé koleno,
13. levý kotník a
14. hrudník.

O detekci kloubů těla se stará třída `parts_detector`. Ta dostane pro detekci snímek, rámečky atletova těla v něm a maximální vzdálenost, o kterou může posunout střed okna od středu atletova těla, což je střed nejmenšího rámečku, do něhož se vejdu rámečky atletova těla. Maximální vzdálenost je určena průměrnou vzdáleností dvojcí středů rámečků atleta.

Detektor si pamatuje poslední velikost kostry, posun středu okna a úhel naklonění trupu. Detekci zkouší na natočení proti poslednímu známemu úhlu trupu a následnému natočení o 20 stupňů oběma směry. Tyto detekce probíhají postupně a vybere se ta, která detekuje nejvíce kloubů těla. Pokud nějaká detekce uspěje na 100 %, další se nezkouší.

Úhel naklonění trupu určuji jako úhel mezi úsečkou spojující střed kyčlí a hlavu a vertikálou, která prochází středem kyčlí. Úhel reprezentuje míru natočení trupu oproti vzpřímené pozici po směru hodinových ručiček. [OBRÁZEK] Hodnoty jsou v rozmezí $[-180, 180]$ stupňů. Metoda `cv::rotate` otáčí snímek o úhel proti směru hodinových ručiček, tudíž bude ve otočeném snímku atletův trup vzpřímený, když mu předám nezměněnou hodnotu úhlu naklonění trupu.

Výpočet úhlu naklonění trupu provádím následující metodou:

```
std::optional<double> get_vertical_tilt_angle(
    const frame_point &a,
    const frame_point &b) noexcept {
    if (a && b) {
        cv::Point2d v1(0, -1); // Point straight up.
        cv::Point2d v2 = *b - *a;
        double dot = v1.x * v2.x + v1.y * v2.y;
        double val = dot / (cv::norm(v1) * cv::norm(v2));
        double mult = 1;
        if (v2.x != 0)
            mult = v2.x / std::abs(v2.x);
        return mult * std::acos(val) * 180.0 / M_PI;
    }
    return std::nullopt;
}
```

Prvním argumentem je pozice středu kyčlí, druhým pozice hlavy atleta. Pro výpočet využívám vlastnosti skalárního součinu, díky kterému lze určit úhel mezi vektory. Při výpočtu vycházím ze vzorce $\langle u, v \rangle = \|u\| \cdot \|v\| \cdot \cos \alpha$, kde α je úhel, který vektory svírají.

Pro detekci kloubů těla nejprve vyberu rámeček okna pro detekci. Jeho střed získám ze středu rámečků atleta a posunu středu z minulého snímku. Rámečky atleta aktualizuju každé 2 snímky, tudíž jejich posunutí při aktualizaci není velké. [OBRÁZEK] Velikost jeho strany určím z velikosti dosud největší detekované kostry, kterou zvětším koeficientem 1.3. Pokud je velikost kostry dosud neznámá, zvětším 1.3 krát nejmenší rámeček obsahující rámečky atletova těla. Škálování rámečků provádím tak, abych zachoval jejich střed a aby se vešly do snímku.

Následně kolem středu zvoleného rámečku okna otočím snímek proti poslednímu známému naklonění trupu, k tomuto úhlu případně příčtu úhel dalšího naklonění, pokud se detekce nepodaří napoprvé. K natočení snímku využiji metodu `cv::rotate`. Z natočeného snímku vyříznu část, která je určena oknem. Střed okna zůstal po rotaci na stejném místě, takže získám jen natočené okno. [OBRÁZEK] Toto okno zmenší na rozlišení, jehož menší strana má 150 px, aby detekce probíhala rychleji. Toto zmenšení nemá příliš velký vliv na přesnost detekcí, ale značný vliv na rychlosť analýzy videa [ODKAZ na experiment].

Zmenšené okno předám jako vstup konvoluční sítě a spustím detekci. Z výsledku detekce extrahuji klouby těla do nezměněného okna, ty podle jeho pozice a naklonění posunu na správnou pozici do snímku.

Nakonec aktualizuji velikost dosud největší detekované kostry, pokud detekuji všechny části těla a tato kostra je největší. Také aktualizuji posun středu okna oproti středu rámečku ohraničujícího atleta. Tento posun zmenším, pokud je větší, než maximální vzdálenost. Jako poslední aktualizuji úhel naklonění trupu, pokud se od posledního známého liší o míň, než 30 stupňů. Tento úhel určuje střed kyčlí a hlava atleta vůči vertikále, pokud byly tyto části těla v předchozím snímku detekovány, jinak úhel neaktualizuju.

Sítě K detekci kloubů těla používám konvoluční síť, která je implementovaná ve frameworku Caffe (Jia a kol., 2014). Tuto síť načtu ze souborů metodou `cv::dnn::readNet`. Váhy sítě a její parametry jsem použil z projektu OpenPose (Cao a kol., 2021). Vstupem této sítě může být libovolně velký obrázek. Výstupem sítě je 4-dimenzionální matice `cv::Mat`, dimenze popisují

- číslo snímku,
- číslo výstupní vlastnosti,
- výšku výstupu a
- šířku výstupu.

Extrakce kloubů z výstupu Číslo snímku je irrelevantní, jelikož předávám síti vždy jen jeden snímek. Výstupních vlastností je pro MPII model 44, ale používám jen prvních 15, které reprezentují klouby těla. Pro lepší přesnost detekce by bylo vhodné využít i zbylé vlastnosti. Z výstupu dostanu matici pravděpodobností kloubu n konstruktorem `cv::Mat`, kterému dám výšku, šířku, typ prvků a ukazatel na výslednou matici, který dostanu z výstupu sítě. Ze získané matice vyberu pozici prvku, který má maximální pravděpodobnost. Pokud je tato pravděpodobnost větší než 0.1, určím přeskálováním pozice bodu ve výstupu poměrem velikosti nezmenšeného vstupu a výstupu sítě pozici bodu ve vstupu. Výslednou pozici určím výpočtem $(x,y) = (x_{out} \cdot w_{in}/w_{out}, y_{out} \cdot h_{in}/h_{out})$. (x_{out},y_{out}) je pozice na výstupu, (w_{in},h_{in}) velikost vstupu a (w_{out},h_{out}) velikost výstupu.

Podle pozice a natočení okna ve snímku určím pozici výsledných bodů ve snímku. Bod posunu o levý horní roh okna, tím získám pozici bodu v otočeném snímku. Otočením bodu podle středu okna o úhel natočení okna zpět získám pozici bodu v nerotovaném snímku. Pro otočení bodu využiji metodu `cv::transform`.

[OBRÁZEK]

Pozice kloubů reprezentuji jako instance `cv::optional<cv::Point2d>`, tedy pozice kloubů, které se mi nepodaří nadetekovat, nejsou definované.

3.5.4 Konverze kostry do 3D modelu

O převod kostry do 3D modelu se stará konstruktor třídy `model`. Konstruktor dostane instanci třídy `person`, která reprezentuje atleta a cestu ke zpracovanému videu.

Nejprve se převedou klouby kostry do 3D, aby se daly body sčítat. Výsledkem konverze bodu (x,y) je bod $(x,0,y)$. Stejná operace se provede s pozicemi levého horního rohu jednotlivých snímků.

Následně se díky těmto výsledkům určí reálná pozice bodů. Mám tedy kloub kostry (x,y,z) a posun snímku (x',y',z') . Výsledná pozice je $(x + x', y + y', z + z')$, pokud atlet běží doprava, $(-x - x', y + y', z + z')$ pokud atlet běží doleva.

Posunutí počátku systému do pozice kotníku odrazové nohy v momentu odrazu se provádí po analýze parametrů odečtením této hodnoty.

3.5.5 Analýza parametrů

Pro analýzu parametrů skoku používám pozice kloubů těla v každém snímku. Tyto hodnoty mám uložené ve vektoru vektorů, jehož prvky jsou instance `std::optional<cv::Point3d>`. První dimenze určuje číslo snímku, druhá číslo kloubu. I když atleta nedetekuju v celém videu, mám pro každý snímek uloženy pozice kloubů. Pokud jsem v daném snímku kloub nedetkoval, je jedna hodnota nedefinovaná (`std::nullopt`).

Detekce kloubů těla není úplně přesná, často při detekci prohodí levá a pravá noha, tuto chybu je potřeba brát při analýze parametrů v potaz. [GRAF levého kotníku, pravého kotníku a nižšího kotníku]

O analýze parametrů se stará třída `vault_analyzer` a struktury reprezentující jednotlivé parametry, pro analýzu používám souřadný systém 3D prostoru.

Důležité momenty skoku

Start Start rozběhu je snímek, ve kterém se začne pohybovat atletův kotník. Postupně zkoumám pozice kotníků ve všech snímcích videa a udržuji si předešlou pozici levého a pravého kotníku. Jakmile se pozice levého nebo pravého kotníku změní o více než 1 pixel, ukončím procházení a vrátím číslo snímku, ve kterém pohyb začal - jedná se předposlední snímek, který jsem zkoumal. Změna pozice kotníků, kterou určuji při zpracování prvního snímku není definovaná, tudíž zpracuji vždy alespoň 2 snímky a výsledné číslo snímku je validní.

O určení momentu startu rozběhu se stará metoda `vault_analyzer::find_start`:

```
std::optional<std::size_t> find_start(const model_video_points &points)
noexcept {
    std::size_t index = 0;
    model_point left = std::nullopt;
    model_point right = std::nullopt;
    for (const auto &body : points) {
        std::optional<double> dist =
            distance(body[body_part::l_ankle], left);
        if (dist && *dist > 1) break;
        dist = distance(body[body_part::r_ankle], right);
        if (dist && *dist > 1) break;
        left = body[body_part::l_ankle];
        right = body[body_part::r_ankle];
        ++index;
    }
    if (index && index != points.size())
        return index - 1;
    return std::nullopt;
}
```

Funkce `distance` funguje stejně, jako metoda `cv::norm` aplikovaná na rozdíl argumentů, pokud jsou oba argumenty validní, jinak vrátí `std::nullopt`.

Odraz Moment odrazu je snímek, ve kterém dochází k odrazu od země při posledním detekovaném kroku. Popis detekce kroků popisují v následující sekci.

Kulminace Moment kulminace nad laťkou je určen snímkem, ve kterém je pozice středu kyčlí nejvýš v celém videu.

Doba trvání jednotlivých kroků

Pro určení doby trvání kroků je potřeba určit specifický moment každého kroku. Zvolil jsem určení momentu odražení od země jednotlivých kroků, tato implementace mi pomohla s detekcí momentu odrazu.

Jelikož si nemohu být jistý pozicí levé a pravé nohy, musím zkoumat pozici nižšího kotníku. Postupně procházím pozici nižšího kotníku v jednotlivých snímcích videa. Určím momenty, ve kterých nižší kotník opouští lokální minimum.

Jelikož detekce nemusí být vždy přesná, musím určit práh, jehož překonání znamená verikální posunutí kloubu kostry. Velikost tohoto prahu musí záviset na velikosti těla atleta a frekvenci snímků videa. Velikost prahu určuji takto: $x \cdot s/fps$, kde s je vzdálenost dvou nejvzdálenějších kloubů těla atleta, fps frekvence snímků videa a x vhodná hodnota. Po analýze vertikálních pohybů nižšího kotníku v momentech odrazů jednotlivých kroků několika videí, jejichž frekvence snímků byla 60 fps, jsem hodnotu x nastavil na $0.023 \cdot 60 = 1.38$. Zvolil jsem nejmenší vertikální posun v momentech odrazů jednotlivých kroků v analyzovaných videích. Jednalo se o videa 1.MOV, 8.MOV a 12.MOV.

Možným vylepšením by bylo zkoumání levé a pravé nohy a jejich vzájemné pozice - která je vepředu a která vzadu - ale jelikož není detekce levé a pravé nohy občas správná, musel jsem se rozhodnout pro výše popsanou implementaci.

Pro přesnost určení kroků je potřeba vyfiltrovat snímky, ve kterých nedochází ke kroku, příkladem je chvíle po odrazu, při níž atlet švihá odrazovou nohou dopředu. Odrazová noha se tedy po odrazu pohybuje vzhůru a následně dolů (pri švihu), což může být z hlediska algoritmu vnímáno jako krok. [OBRÁZEK]

Budu postupně procházet snímky, ve kterých nižší kotník opouští lokální minimum, označím je jako snímky odrazů. Mezi následujícími snímky odrazů najdu nejvyšší pozici nižšího kotníku. Následující snímek odrazu označím za krok jen v případě, že se v něm nižší kotník nachází pod průměrem výšky nalezeného maxima nižšího kotníku a výšky nižšího kotníku v momentu předešlého odrazu. První snímek odrazu ve výsledném seznamu nechám. [OBRÁZEK]

Abych nedetekoval kroky po provedení skoku, ukončím filtrace (a další kroky do výsledku nepřidám) jakmile narazím na snímek, který reprezentuje odraz při kroku a nižší kotník se v něm nachází výš, než je nejvyšší pozice nižšího kotníku mezi prvním a druhým krokem. [OBRÁZEK]

Podle frekvence snímků videa a vyfiltrovaných čísel snímků, v nichž dochází k odrazu při krocích, určím dobu trvání kroků. Doba trvání i -tého kroku je tedy $d_i = (k_{i+1} - k_i) \cdot fps$, k_i je číslo snímků i -tého odrazu kroku a fps je frekvence snímků videa.

Ztráta rychlosti ramen a boků

Pro výpočet ztráty horizontální rychlosti ramen a boků využívám střed ramen a kyčlí atletova těla. Nejprve určím pozici středu boků 0.1 s před momentem odrazu, při odrazu a 0.1 s po momentu odrazu.

Pokud byly úspěšné detekce ve všech popsaných momentech a mám jistotu, že nebudu dělit nulou, spočítám výslednou ztrátu rychlosti vzorcem $l = (1 - (a - d)/(d - b)) \cdot 100$, kde a je hodnota první složky pozice zkoumané části těla 0.1 s po odrazu, d určuje stejnou hodnotu v momentu odrazu a b před odrazem. Výslednou ztrátu určuji v procentech.

Pokud hodnotu výrazu nelze spočítat, má tento parametr nedefinovanou hodnotu.

Výška boků

V průběhu celého videa určuji výšku boků. Pro každý snímek uložím hodnotu průměru výšky levé a pravé kyčle. Pokud se ve snímku podařilo detektovat jen jednu kyčli, uložím výšku detekované, pokud se nepodaří detektovat ani jednu kyčel ve snímku, je hodnota výšky boků v tomto snímku nedefinovaná.

Úhel odrazu

Úhel odrazu určuji jako rozdíl úhlu pohybu na začátku skoku a úhlu pohybu na konci rozběhu. Úhly pohybu udávají úhel mezi x -ovou osou a směrem pohybu středu kyčlí atleta.

Pro výpočet úhlu odrazu získám pozici středu kyčlí 0.1 s před momentem odrazu, při odrazu a 0.1 s po odrazu. Z pozice kyčlí při odrazu a po něm určím úhel, pod kterým se pohybuje atlet při skoku. Na základě pozice kyčlí při odrazu a před ním vypočítám úhel pohybu před odrazem.

Kladná hodnota úhlu odrazu znamená odraz vzhůru, záporná odraz dolů. Hodnoty jsou v intervalu $[-180, 180]$ stupňů.

Úhel došlapu jednotlivých kroků

Algoritmem, kterým získávám momenty došlapů pro výpočet oporové fáze kroků, získám momenty došlapů všech kroků. V nich určím úhel mezi vertikálou a úsečkou, kterou určuje kotník a kyčel došlapující nohy.

Jedná se o míru došlapu nohy před kyčel, při došlapu před kyčel je tedy hodnota kladná, při došlapu za kyčel je hodnota záporná. Hodnoty se pohybují mezi -180 a 180 stupni.

Náklon trupu

Náklon trupu je úhel mezi vertikálou a úsečkou, kterou určuje pozice středu kyčlí a pozice hlavy. Tento úhel vypočítám pro každý snímek videa.

Úhel určuji stejně, jako v případě detekce kostry (sekce 3.5.3), jen se jedná o úhel v trojrozměrném prostoru. Jedná se o míru naklonění, jejíž hodnota je kladná, pokud je atlet v předklonu a záporná, pokud je v záklonu. Směr rozběhu je ve směru rostoucí x -ové souřadnice.

Hodnota naklonění trupu se pohybuje v rozmezí -180 až 180 stupňů.

3.6 Analýza modelu

O načtení modelu ze souboru se stará třída `model`, podle uložených pozic kloubů kostry vůči levému hornímu rohu snímku a posunu snímku ve videu se klouby převedou do 3D modelu. Tento postup je stejný jako při převodu detekované kostry do 3D modelu. Tento model je rozdíl od konverze z detekované kostry posunut, aby byl počátek systému v místě kotníku odrazové nohy v momentu odrazu.

Pro následnou analýzu parametrů se využívá model ve 3D souřadném systému stejným způsobem, jako při analýze videa.

3.7 Výstup programu

3.7.1 Uložení výstupu

Výstup ukládám do textových souborů, které jsou uloženy ve složce `outputs/name/`, kde `name` je název zpracovávaného videa (bez cesty a přípony). Parametry ukládám do souboru `parameters.csv`, model do souboru `model.txt`.

Uložení parametrů provádí třída `vault_analyzer`, uložení modelu má na starosti třída `model`. Pro vypisování hodnot do souboru používám `std::ofstream`.

Struktura těchto souborů je popsán v sekci 2.5.1.

Kromě těchto textových souborů generuje program trojici videí, kterou uloží do stejné složky jako textové soubory.

Prvním z nich je video bez detekcí, obsahuje původní snímky videa, která program zmenší před analýzou. Název tohoto videa je `raw.avi`.

Druhé video se generuje jen v případě, že uživatel zvolí automatickou detekci atleta, obsahuje postavy, které program v každém videu bral v potaz při hledání atleta. Jedná se o video `found.avi`.

Třetí video obsahuje zakreslenou kostru, rámečky těla atleta a pozici trasovaného pozadí. Toto video se zapíše do souboru `detections.avi`.

3.7.2 Zobrazení výstupu uživateli

O funkcionalitu zobrazení výstupu uživateli se stará třída `viewer`. Ta nejprve otevře prohlížeč snímků videa a následně zobrazí grafy obsahující hodnoty parametrů.

Prohlížeč snímků

Prohlížeč snímků spustím zavolením metody `viewer::show`. Jejimi parametry jsou snímky videa s detekcemi, bez detekcí a instance třídy `vault_analyzer`, která analyzovala atletův pohyb.

Prohlížeč si udržuje číslo právě zobrazovaného snímku, při jeho změně kontroluje, aby se jeho hodnota nedostala mimo hodnoty, kterými lze přistupovat k jednotlivým snímkům.

V nekonečné smyčce provádí program následující kroky:

Snímek zobrazím metodou `cv::imshow`, která vytvoří okno a v něm zobrazí snímek podle čísla právě zobrazovaného snímku.

Spolu se zobrazením snímku vypíše program do konzole číslo snímku, dobu snímku vůči odrazu a hodnoty parametrů, které s daným snímkem souvisí. [TABULKA]

Následně se čeká na vstup uživatele s využitím metody `cv::waitKey`. Tato metoda vrací kód klávesy, kterou uživatel stiskl a čeká na vstup uživatelského neomezeně dlouhou dobu.

Levou a pravou šipkou se uživatel pohybuje mezi snímky videa dopředu, případně dozadu, mezerníkem uživatel zobrazí a opět skryje zakreslené detekce kostry atletova těla. Stiskem klávesy `esc` se zobrazí grafy s hodnotami parametrů a ukončí se prohlížení snímků videa přerušením smyčky.

Grafická reprezentace parametrů

Zobrazení parametrů v grafické podobě zajišťuje skript `plot_parameters.py`, který načte hodnoty parametrů ze souboru, který se vygeneroval při analýze parametrů.

Skript se spouští přímo z kódu jazyka C++, což zajišťuje API pro použití Pythonu prostředky, kterými disponuje jazyk C (Foundation, 2021). O spouštění se stará třída `viewer`.

Kód, který spouští skript pro zanesení parametrů do grafů vypadá následovně:

```
void show_parameters() const noexcept {
    Py_Initialize();

    FILE *file = fopen("plot_parameters.py", "r");
    if(file) {
        wchar_t *argv[3] = {
            Py_DecodeLocale("plot_parameters.py", NULL),
            Py_DecodeLocale("--file", NULL),
            Py_DecodeLocale(params_filename.c_str(), NULL)
        };
        PySys_SetArgv(3, argv);
        PyRun_SimpleFile(file, "plot_parameters.py");
        fclose(file);
    }

    Py_Finalize();
}
```

4. Experimenty

4.1 Volba úvodní pozice atleta

4.1.1 Pozice ohraničujícího rámečku

Posunutí úvodního ohraničujícího rámečku.

4.1.2 Velikost ohraničujícího rámečku

Různé velikosti úvodního ohraničujícího rámečku.

4.2 Pohyb kamery

4.2.1 Vertikální pohyb kamery

Vliv pohybu kamery na hodnoty parametrů.

4.2.2 Rotace kamery

Rotace kamery po a proti směru hodinových ručiček.

4.3 Pozice kamery

4.3.1 Směr rozběhu

Porovnání analýzy skoku z různých stran.

4.3.2 Úhel záběru

Porovnání analýzy skoku z různých úhlů.

4.3.3 Vzdálenost kamery

Porovnání analýzy skoku z různé vzdálenosti od rozběžiště.

4.4 Další postavy ve videu

Vliv ostatních postav na detekci.

4.5 Frekvence snímků

Analýza videa po vypuštění snímků.

4.6 Závody

Analýza videí ze závodu.

4.7 Shrnutí

Shrnutí analýzy testovacích videí.

Závěr

Seznam použité literatury

- ANDRILUKA, M., PISHCHULIN, L., GEHLER, P. a SCHIELE, B. (2014). 2D Human Pose Estimation: New Benchmark and State of the Art Analysis. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- ATHLETICS, W. (2021). Research centre. [online]. URL <https://www.worldathletics.org/about-iaaf/documents/research-centre>. Naposledy navštívěno 17. 5. 2021.
- BRADSKI, G. (2000). The OpenCV Library. *Dr. Dobb's Journal of Software Tools*.
- CAO, Z., HIDALGO, G., SIMON, T., WEI, S.-E. a SHEIKH, Y. (2021). OpenPose: Realtime Multi-Person 2D Pose Estimation Using Part Affinity Fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **43**(1), 172–186. doi: 10.1109/TPAMI.2019.2929257.
- FOUNDATION, P. S. (2021). 1. Embedding Python in Another Application. [online]. URL <https://docs.python.org/3.8/extending/embedding.html>. Naposledy navštívěno 23. 5. 2021.
- GRAVESTOCK, H., BISSAS, A. a MERLINO, S. (2017). Biomechanical report for the IAAF World Championships London 2017: Pole Vault Men's. URL <https://www.worldathletics.org/download/download?filename=e2903e25-0ee3-43d2-b031-1af5a3b84fb8.pdf&urlslug=Men%27s%20pole%20vault%20-%202017%20IAAF%20World%20Championships%20Biomechanical%20report>.
- HUNTER, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*, **9**(3), 90–95. doi: 10.1109/MCSE.2007.55.
- JIA, Y., SHELHAMER, E., DONAHUE, J., KARAYEV, S., LONG, J., GIRSHICK, R., GUADARRAMA, S. a DARRELL, T. (2014). Caffe: Convolutional Architecture for Fast Feature Embedding. *arXiv preprint arXiv:1408.5093*.
- LUKEŽIČ, A., VOJÍŘ, T., ČEHOVIN ZAJC, L., MATAS, J. a KRISTAN, M. (2018). Discriminative Correlation Filter Tracker with Channel and Spatial Reliability. *International Journal of Computer Vision*, **126**(7), 671–688. ISSN 1573-1405. doi: 10.1007/s11263-017-1061-3. URL <http://dx.doi.org/10.1007/s11263-017-1061-3>.
- MATLAB (2010). *version 7.10.0 (R2010a)*. The MathWorks Inc., Natick, Massachusetts.
- ROSEBROCK, A. (2015). (Faster) Non-Maximum Suppression in Python. [online]. URL <https://www.pyimagesearch.com/2015/02/16/faster-non-maximum-suppression-python/>. Naposledy navštívěno 21. 5. 2021.
- WILLIAMS, T., KELLEY, C. a MANY OTHERS (2013). Gnuplot 4.6: an interactive plotting program. [online]. URL <http://gnuplot.sourceforge.net/>.

Seznam obrázků

2.1	Ohraničující rámeček.	11
2.2	Ztráta atleta při použití různých trackerů.	13
3.1	UML diagram zobrazující strukturu programu.	21

Seznam tabulek

Seznam použitých zkratek

A. Přílohy

A.1 První příloha