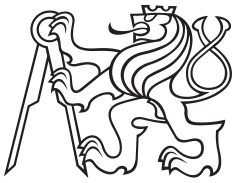
Diplomová práce



České
vysoké
učení technické
v Praze

F3

Fakulta elektrotechnická
Katedra kybernetiky

Grafický doprovod tanečního představení

Bc. Ondřej Tyle

Vedoucí: Ing. Berka Roman Ph.D.
Obor: Kybernetika a robotika
Květen 2022

Poděkování

Poděkování patří hlavně vedoucímu práce Ing. Romanu Berkovi Ph.D. za rady a podporu, dále také choreografce Adéle Kašparové za spolupráci při tvorbě celého projektu a skvělé zážitky při jeho prezentaci a také všem účinkujícím tanečníkům. Za technickou podporu pak děkuji Ing. Ondřejovi Slabému.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně, a že jsem uvedl veškerou použitou literaturu.

V Praze, 20. května 2022

Abstrakt

Tato práce se zabývá tvorbou systému umožňujícího promítání 2D grafiky při tanečním vystoupení, která je schopná reagovat v reálném čase na pohyby tanečníků.
V první části je shrnutí současných metod detekce lidské postavy. Následně je uveden teoretický návrh celého systému se všemi jeho komponenty. V další části je podrobný technický popis implementace všech komponent. V poslední části je uveden soupis všech animací a komponent použitých při vystoupení včetně jejich implementace.

Klíčová slova: detekce polohy lidského těla, projekce, geometrie, shadery, GLSL, Arcade

Vedoucí: Ing. Berka Roman Ph.D.

Abstract

This work deals with the creation of a system that allows the projection of 2D graphics during a dance performance, which is able to respond in real time to the movements of dancers.
The first part is a summary of current methods of human pose estimation. Subsequently, the theoretical design of the whole system with all its components is given. The next section provides a detailed technical description of the implementation of all components. The last part contains a list of all animations and components used in the performance, including their implementation.

Keywords: human pose estimation, projection, geometry, shaders, GLSL, Arcade

Title translation: Graphic accompaniment of a dance performance

Obsah

Obrázky

Úvod

Grafický doprovod se v tanečních a divadelních vystoupeních objevuje stále častěji. Pod tímto pojmem se skrývá jakákoli světelná projekce. Může se tedy jednat jen o správné nasvícení, nebo také o složité animace a videomapping[1]. V současné době se stále více experimentuje s projekcí "reagující" na pohyb tanečníků. Jedná se o předem připravené animace, které se správně vytvořenou choreografií vypadají, že reagují na pohyb tanečníků. Vznikají ale také systémy, které opravdu reagují na taneční pohyby. K tomu je potřeba nejen promítat vytvořenou grafiku, ale také získat informaci o poloze tanečníků.
Cílem této práce je seznámení se se současnými možnostmi projekce grafiky při vystoupeních a se současnými metodami detekce polohy celé osoby i částí jejího těla. Praktickou částí je pak vytvoření systému, který dokáže vykreslovat 2D animaci až na dvě místa současně, snímat polohu tanečníků po celé ploše jeviště a pomocí těchto dat vykreslovat animace reagující na tanečníky v reálném čase.
Práce je tvořena současně s bakalářskou prací choreografky Adély Kašparové z HAMU. Při řešení jsou kombinovány požadavky technické i choreografické.

[1] Videomapping je směr vizuálního umění, které využívá projekci ve volném prostoru na libovolné objekty, např. fasády domů nebo interiéry budov.

Kapitola 1

Historie a současný stav

Současná technologická úroveň jde nesmírně rychle kupředu. To co by před sto lety trvalo vymyslet nebo vypočítat rok, dnes zvládneme za pár hodin. O stávajících technologických vymoženostech jsme si mohli ještě před pár lety nechat jen zdát. Stejně tak tomu je i s projekcí ve vystoupeních, nehledě na možnosti mapování lidského těla.

1.1 Počátky tanečních vystoupení s projekčními prvky

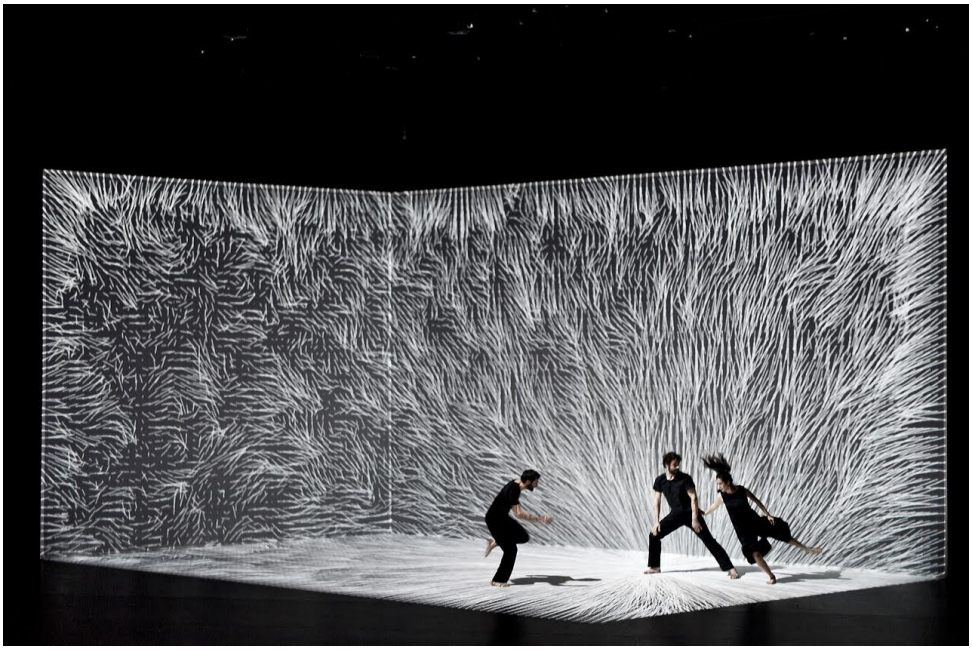
Divadelní a taneční představení se již od začátku jejich tvorby začala obohacovat o kostýmy a další prvky, které by je nějakým způsobem oživily a dokreslily jejich atmosféru. Nejprve se jednalo jen o jednoduché kostýmy a rekvizity, ale s přibývajícím se časem a stále vyspělejší technologií se začalo experimentovat se zapojením moderní techniky.
S osvětlením to bylo obdobné. Nejprve se používalo přírodní sluneční světlo. Poté se začaly využívat svíce a olejové lampy. Po objevení možnosti svítit plynem se do divadel začaly instalovat plynové reflektory. V roce 1878 si Joseph Swan nechal patentovat svou první elektrickou lampu a od té doby začala všude po světě vznikat elektrická osvětlení.

Roku 1958 se v Bruselu konala mezinárodní výstava *Expo 58*, na které *Alfréd Radok* a *Josef Svoboda* předvedli celosvětově unikátní a nový koncept sjednocení jevištního vystoupení a filmové projekce. Jejich snaha byla, aby projekce nebyla jen pohyblivou kulisou, ani nevytvářela pouze zdání skutečnosti. Chtěli propojit obsah dějů na scéně s akcí na filmovém plátně.

Obrázek 1.1: Vystoupení *Laterna magika* na výstavě *Expo 58*. (převzato z [24])

V posledním desetiletí vznikají po celém světě unikátní představení, která spojují vystoupení tanečníků s videomappingem a projekcí. Jako příklad můžeme uvést vystoupení *The Movement of Air* [20] od studia *Adriena M & Claire B*, nebo také scéna v projektu *Verses / SKETCH 9: Perspectives* [25].

Největším problémem současných vystoupení s projekcí je synchronizace pohybů tanečníka s hudbou a zároveň s promítanou grafikou, což velmi zvyšuje náročnost celé přípravy vystoupení.



Obrázek 1.2: Vystoupení *The Movement of Air*. (převzato z [20])

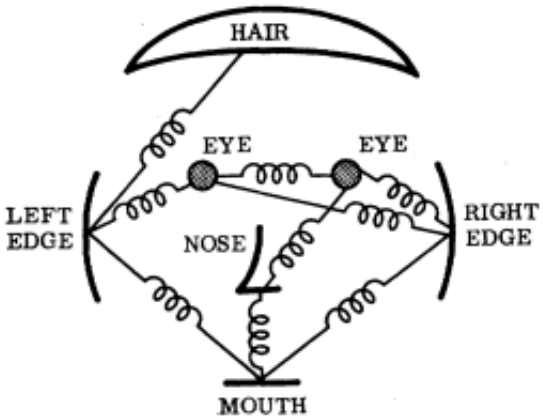
Obrázek 1.3: Vystoupení *Perspectives*. (převzato z [25])

Pokud bychom místo učení tanečníka na danou projekci tento problém obrátili, mohl by se tanečník pohybovat podle dané choreografie, ale nemusel by se učit přesné pohyby. Animace by pak reagovala na to kde se nachází a vykreslovaná grafika by vždy byla s jeho pohyby synchronní.

1.2 Počátky a současné možnosti detekce lidského těla

Detekce lidského těla a jeho pohybu je ve virtuálním světě ovlivněna mnoha faktory, které ji komplikují. Je to nejen samotná složitost lidského těla jako mechanického objektu (230 kloubů, 244 stupňů volnosti), ale i řada dalších, od rozdílné stavby lidského těla různých jedinců, přes rozdíly v oblečení, až po faktory spíše technické, jako je vliv osvětlení, rozlišení kamery, nebo výkon hardwaru.

"Pružinový"(spring) model:
Základní myšlenku, jak popsat lidské tělo, vyslovil v roce 1973 Martin A. Fischler ve svém článku *The Representation and Matching of Pictorial Structures* [1] a to takzvaným "pružinovým modelem" (spring model). Tento model spočívá v rozdělení těla na jeho důležité části a jejich propojením pružinami. Tyto pružiny mají své limity, aby se nemohlo stát, že se například noha člověka bude otáčet o 365 stupňů.

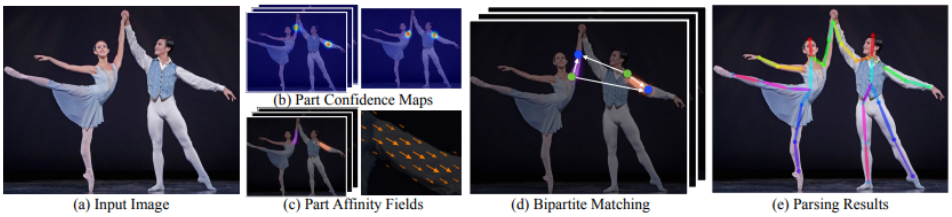


Obrázek 1.4: Znázornění hlavy podle článku [1] M. A. Fischlera.

Tato myšlenka je základem, ze kterého dnes vycházejí mnohem složitější modely. Například v roce 2011 ve svém článku *Articulated pose estimation with flexible mixtures-of-parts* [2] Yi Yang a D. Ramanan vytvořili model "flexibilní směsi"(flexible mixture model), ve kterém navíc využívají lokální tuhosti.

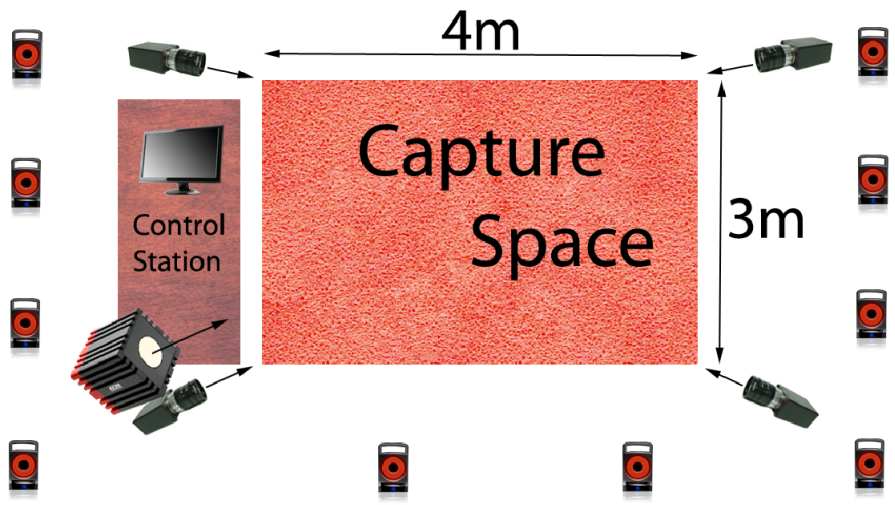
Strojové učení:
Od roku 2016 se do popředí dostalo strojové učení a metody jeho hlubokého učení se staly dominantními v rámci odhadu 2D i 3D pozic kloubů a 3D tvaru těla z velkých trénovacích sad.

2D pozice kloubů:
První modely hlubokého učení se zaměřily na extrakci 2D pozic lidských kloubů z obrázku. Tyto modely daný snímek provedou konvoluční neuronovou sítí, aby získaly řadu tepelných map (jednu pro každý kloub), které nabývají vysokých hodnot tam, kde jsou klouby detekovány.
Při detekci více lidí najednou musíme rozhodnout, jak rozlišit, komu detekované body těla patří. Tento problém můžeme řešit dvěma variantami. Buďto budeme body detekovat odspodu a zároveň je přiřazovat k daným končetinám, nebo budeme body detekovat shora a nejprve na fotku použijeme neuronovou síť, která detekuje jednotlivé postavy a až poté na detekované postavy použijeme neuronovou síť pro detekci částí těla.
Například metoda popsaná v článku *Realtime Multi-person 2D Pose Estimation Using Part Affinity Fields* [7] se umístila jako první v soutěži *COCO 2016 keypoints challenge*. Tato metoda detekuje body odspodu a je schopná fungovat v reálném čase bez ohledu na počet lidí na snímku.



Obrázek 1.5: Postup detekce bodů lidského těla podle metody popsané v článku [7]

3D pozice kloubů:
S příchodem více datových sad s lidskou pózou anotovanou ve více pohledech [13] [14] se modely, které detekují 3D polohy kloubů, staly populárnějšími.
Ve článku *Human3.6M: Large Scale Datasets and Predictive Methods for 3D Human Sensing in Natural Environments* [3] je prezentovaný jeden takovýto dataset. Bylo zde nasnímáno pět žen a šest mužů ze čtyř stran. Celkem tento dataset obsahuje 3.6 milionů 3D poloh člověka.

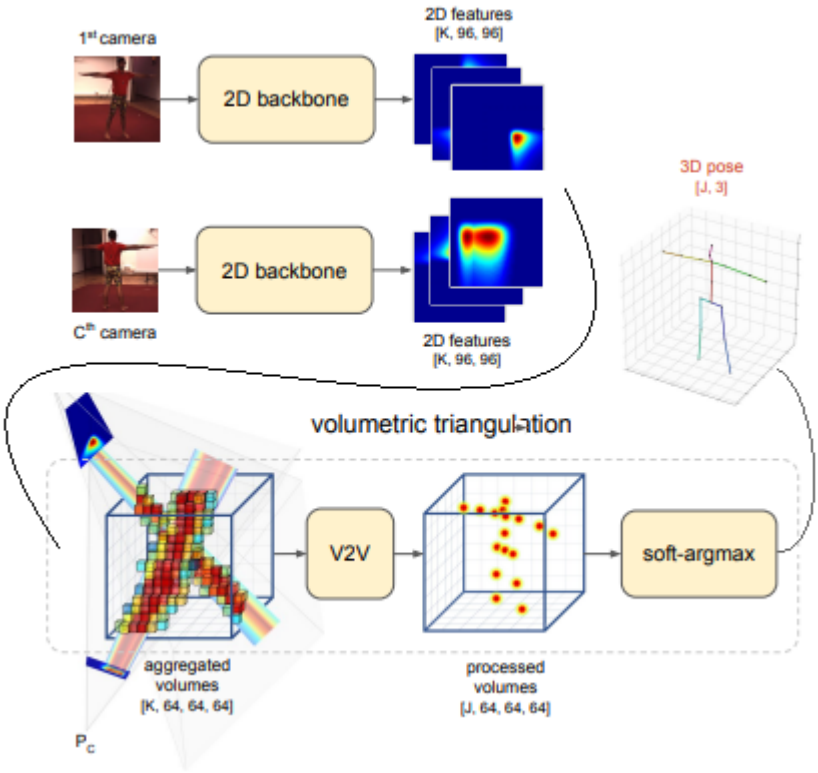


Obrázek 1.6: Schema při pořizování datasetu popsaného v článku [3].

Tyto modely opět spadají do dvou kategorií. V první se neuronová síť používá k detekci 2D kloubních pozic z každého pohledu a tyto detekce jsou pak triangulovány pro získání 3D pozic kloubů. 2D síť může být vylepšena tak, aby produkovala lepší detekce založené na 3D datech. Navíc tyto přístupy často obsahují filtry ve 2D i 3D pro zpřesnění detekovaných bodů. [12]]
Ve druhém je neuronová síť trénována end-to-end, aby předpovídala 3D polohy kloubů přímo ze sady snímků, bez mezilehlých detekcí 2D pozic kloubů. Takové přístupy často promítají obrazové prvky do krychle a poté používají 3D konvoluční neuronovou síť k predikci 3D teplotní mapy pro každý kloub.

3D tvar těla:
Při detekci 3D tvaru těla obecně platí, že pro každé tělo v rámci obrázku jsou detekovány některé klíčové body a silueta a poté jsou parametry 3D modelu tvaru přizpůsobeny tak, aby odpovídaly poloze klíčových bodů a siluetě. Jedním z takovýchto modelů je například *SMPL* uvedený v článku *SMPL: a skinned multi-person linear model* [4].

Využití:
Využití takto získaných bodů je hojné. Dají se takto například vytvářet animace postavy, v medicíně pomáhá detekovat tělesné vady jako je třeba skolióza, nebo se začíná využívat, jak již bylo zmíněno, i při tvorbě umění.



Obrázek 1.7: Metoda detekce 3D poloh kloubů v článku [9].

1.3 Grafický doprovod tanečního představení

Taneční představení je v této práci specifikováno na vystoupení s prvky streetu[1]. Grafickým doprovodem se myslí projekce jednoduchých 2D objektů za a pod tanečníky. Tato animace by měla být choreograficky sladěna s tanečním projevem a měla by obsahovat prvky interakce s tanečníkem v reálném čase. Tanečník by měl být schopen jak svými pohyby spouštět vykreslovanou animaci, tak jí i interaktivně generovat a interagovat s ní v reálném čase. Požadavky na vzhled nejsou nijak náročné. Mělo by se jednat o černobílou projekci základních geometrických útvarů s lehkým prvkem "glow"[2] efektu pro oživení.
Celá práce by měla sloužit jako systém připravený pro použití při dalších vystoupeních. Mělo by se jednat o funkční kostru, která se vždy doplní o dané efekty a animace ve vystoupení používané, které si bude moci uživatel sám jednoduše připravit a následně je bude moci snadno vložit do daného systému a využít při svém představení.

[1] Street dance je zastřešující pojem, který se používá pro popis tanečních stylů, které se vyvinuly mimo taneční studia.
[2] Drobné rozostření, které zjemní obrys objektu, aby nebyl tak kontrastní.

Celková kostra by měla obsahovat možnost vykreslovat až na dvě místa současně, možnost snímat polohu tanečníků v reálném čase a použít data o jejich poloze a o poloze jejich kloubů ke generování animací a dále také vykreslit předem připravené animace.

Kapitola 2

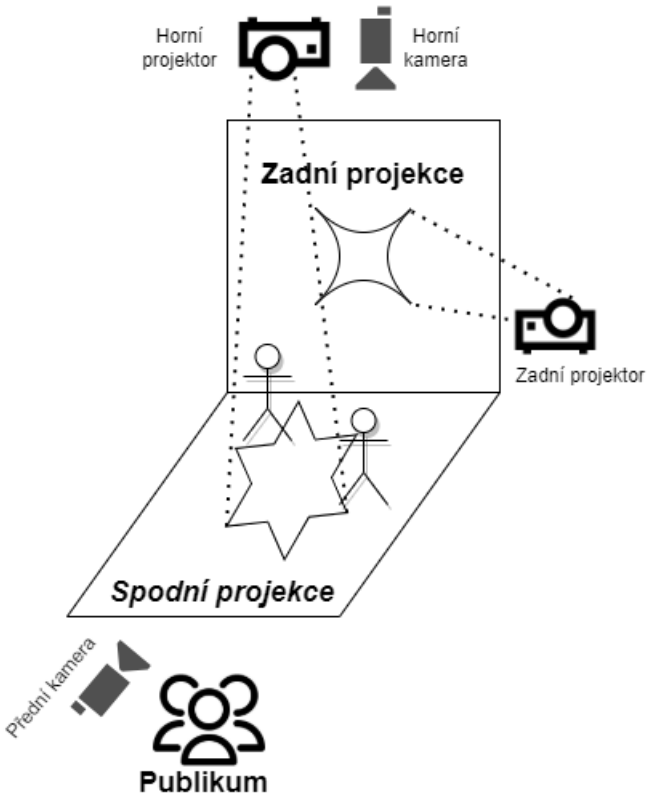
Teoretický rozbor

Při teoretickém návrhu celého projektu se musely skloubit požadavky na choreografii a samotné taneční představení s technickými možnostmi a dostupnými zařízeními. Tato kapitola popisuje návrh celého projektu s jeho jednotlivými komponenty.

2.1 Návrh prostředí

Taneční představení je navrhnuto tak, aby spojovalo prvky tance s vykreslovanou grafikou. Proto bylo potřeba vymyslet, jak nejlépe spojit grafickou projekci s tanečníky. Inspirace byla čerpána převážně z tanečního představení *Pixel* [5], z vystoupení od *Sila Sveta* [6] a z vystoupení *Into the Light* [8]. Společně s choreografkou bylo navrženo schéma na obrázku 2.1. Do schématu byly zakomponovány dva projektory pro horní a zadní projekci a dvě kamery na snímání polohy tanečníků.

Cílem tohoto návrhu bylo pokrýt co největší oblast za a pod tanečníky vykreslovanou grafikou. Pomocí horního projektoru se animace vykresluje pod tanečníky na taneční ploše. Kvůli zabránění projekce na tanečníky byla zvolena zadní projekce na zadní plátno. Pro detekci tanečníků byly umístěny dvě kamery, jedna k hornímu projektoru a druhá ze strany publika.



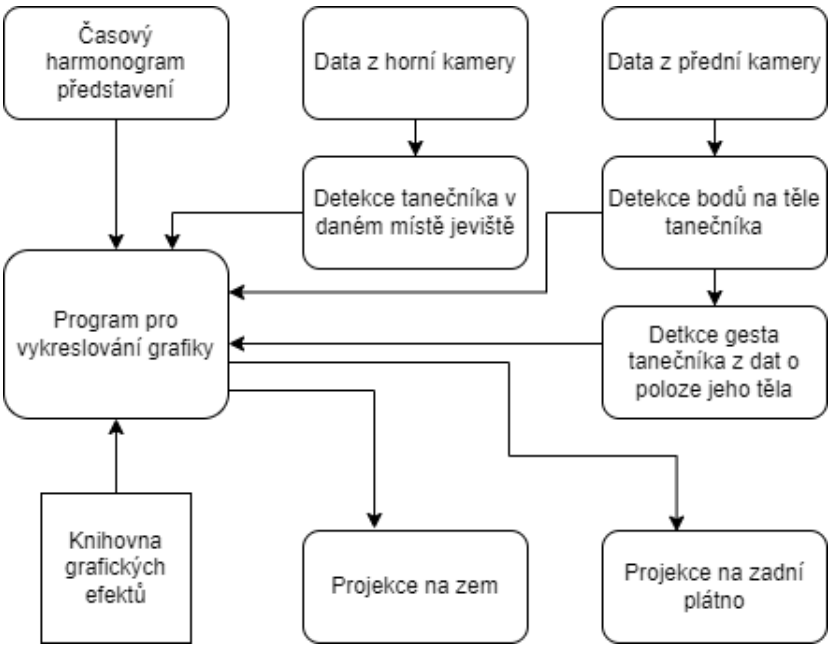
Obrázek 2.1: Schéma prostředí pro vystoupení.

2.2 Struktura procesu vykreslování grafiky

Při návrhu struktury se vycházelo z požadavků na jednotlivé jeho funkce. Bylo potřeba zajistit, aby program mohl současně vykreslovat grafiku na obě projekční plochy, snímat polohu tanečníků a detekovat jejich pohyby (gesta) a bylo zde implementováno něco na způsob časové osy a možnosti definovat časy spouštění jednotlivých animací.

Dalším požadavkem bylo, aby byl program jednoduchý a intuitivní pro jakéhokoli uživatele, tak, aby tento systém mohl používat kdokoli, kdo bude řešit stejný nebo obdobný úkol. Program by proto měl být psaný srozumitelně a pro daného uživatele by mělo být snadné provést potřebné úpravy pro své představení.

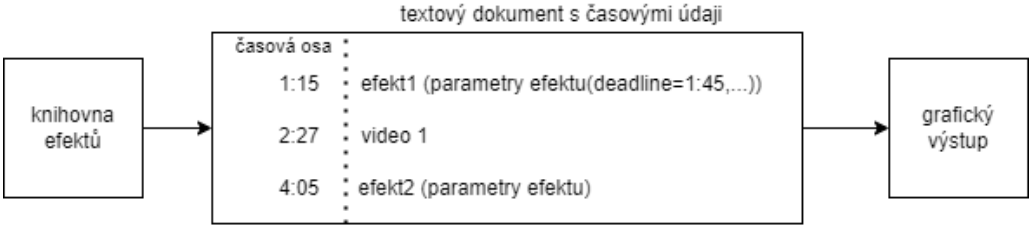
Jak je vidět z návrhu struktury, je zde prvek časového harmonogramu. Tento prvek zastává úlohu řízení projekce při vystoupení. Dále jsou zde také dva prvky na detekci tanečníků pomocí kamery.



Obrázek 2.2: Návrh struktury procesu vykreslování grafiky.

2.3 Řízení představení v čase

Pro co největší srozumitelnost ze strany nového uživatele je potřeba sjednotit všechny řídící prvky do jednoho komponentu. Tento komponent by měl obsahovat vše, čím by uživatel mohl nastavit své vystoupení.
V tomto případě byl zvolen textový soubor, pomocí něhož je možné časově řídit jednotlivé animace a nastavovat jejich parametry.
Při návrhu tohoto souboru bylo uváženo, že uživatel nemusí mít žádné programovací znalosti, a i přesto by měl být schopný drobných úprav, například změnit čas spuštění dané animace.
Soubor byl navrhnut tak, že na každém řádku bude vždy informace o tom, co se bude vykreslovat, kam se to bude vykreslovat a časový údaj ze kterého bude jasný začátek a konec dané animace.
Prvotní návrh struktury takového souboru je vidět na dalším obrázku.



Obrázek 2.3: Návrh komponentu pro řízení představení.

Z obrázku 2.3 je vidět, že v dokumentu bude jasná časová osa, kterou bude možné podle libosti pozměňovat. Toto je důležité při nácviku představení a umožňuje tak jednoduše spouštět vybrané animace podle potřeby nácviku.

2.4 Detekce tanečníka pomocí horní kamery

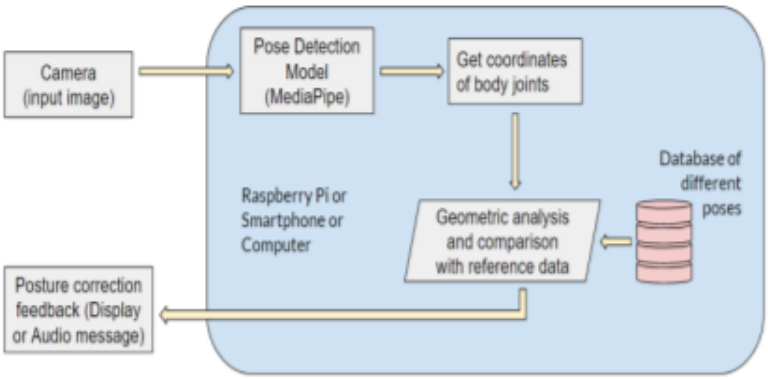
Pro detekci tanečníka v prostoru byl zvolen způsob detekce pomocí kamery umístěné nad taneční plochou. Tato detekce má sloužit pouze pro zjištění přibližné polohy tanečníka v daném území. Toto území je omezeno oblastí viditelnou pro kameru, a také oblastí do které se dá promítat vykreslovaná grafika.
Při návrhu rozložení kamery a projektoru se kladl důraz na co největší přesnost pro výslednou detekci. Přesnost je největší, pokud se kamera nachází přímo nad středem daného území a pokud jsou dostatečné světelné podmínky pro detekci postav.
Výsledkem této detekce je poloha tanečníka v prostoru. Tato poloha se pak použije při vykreslování grafiky reagující na pohyb tanečníka.

2.5 Detekce tanečníka pomocí přední kamery

Některé animace v představení jsou tvořeny tak, že reagují na jednotlivé části těla tanečníka. Tato reakce se nevztahuje na jeho polohu v prostoru, ale na polohu částí jeho těla v prostoru zadní projekce (od země ke stropu).
Pro tento případ detekce musí být kamera umístěna před tanečníkem tak, aby zabírala celé projekční plátno, které je za ním. Pro správné fungování by kamera měla vidět celého tanečníka.
Detekované body jsou zobrazeny s největší přesností, pokud tanečník stojí blízko zadního plátna. S rostoucí vzdáleností se nepřesnost zvětšuje.

2.6 Detekce gest tanečníka

Ve vystoupení se mohou objevit animace, které nejsou spouštěny v daném časovém okamžiku, ale spouští je sám tanečník pomocí naučeného gesta. Toto gesto je vlastně jen poloha jeho těla, například postoj snožmo s rukama podél těla. Komponenta detekující toto gesto má předem definováno, jak takové gesto vypadá. Pokud je detekce spuštěna a komponenta detekuje tanečníka ukazujícího dané gesto, je následně spuštěna vybraná animace.
Způsob detekce by měl fungovat obdobně jako v článku [15], ve kterém se určuje poloha těla při cvičení jógy. Princip detekce je vidět na následujícím diagramu.



Obrázek 2.4: Diagram programu pro detekci gesta převzatý z [15].

Gest zde může být naprogramováno více pro případ, že je na jevišti více tanečníků, nebo pokud se v dané choreografii objevuje jedna nebo více situací, kdy tanečník vykonává dané gesto aniž by chtěl spouštět nějakou animaci.

2.7 Vykreslovaná grafika v představení

Cílem představení je efektivně spojit tanec a animaci. Při návrhu vykreslované grafiky byl kladen důraz na estetiku a jednoduchost. Při vystoupení nechceme, aby byl divák zahlcen všemožnými vjemy a nemohl se rozhodnout, na co se dříve dívat. Chceme naopak, aby se mohl soustředit vždy jen na to podstatné a ostatní prvky vždy jen dotvářely celkový vjem.
Z těchto důvodů byla grafika navrhnuta jako základní geometrické obrazce jako jsou přímka, kružnice, obdélník a trojúhelník. Celková struktura animace pak byla stanovena jako vykreslení bílého objektu na černé pozadí s prvkem "glow" efektu pro sjednocení.

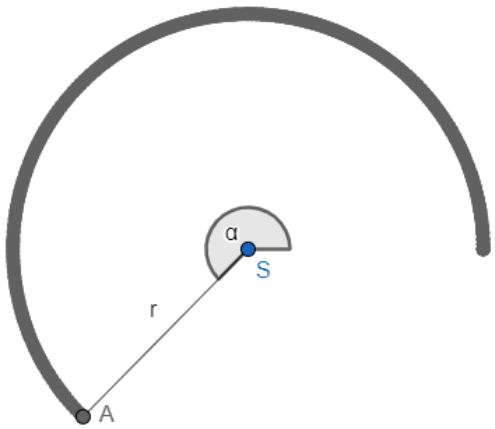
Celé představení je rozděleno do pěti částí. Každá z nich má určený svůj geometrický tvar, který dominuje ve vykreslované grafice. Tyto tvary určují styl dané části představení.Jsou to kružnice, přímka, trojúhelník, obdélník a spirála.
Během vystoupení se zobrazují různé typy grafiky. Prvním a nejjednodušším je předem připravená animace. Ta se používá jako vodící prvek každé části představení a určuje daný geometrický tvar. Takto vytvořená animace je spouštěna většinou na začátku a na konci každé části představení.
Dalším typem grafiky jsou animace reagující na tanečníka. Tyto animace se ještě dělí podle plochy, na kterou se vykreslují. Pokud jde o zem, jedná se většinou o animace reagující na pohyb tanečníka po dané ploše. Jedná-li se však o animaci vykreslující se za tanečníka, animace může být i složitější a může reagovat na jakékoli části těla tanečníka. Dá se zde tedy vytvořit zajímavější a interaktivnější animace, než při projekci na zem.

2.8 Teoretický návrh animací

Při návrhu jednotlivých animací bylo potřeba matematicky navrhnout a popsat chování jednotlivých vykreslovaných objektů. Pokud například chceme vykreslit kruh, je potřeba určit parametry, pomocí nichž bude vykreslovaný. Můžeme určit například střed a poloměr, nebo dva body, mezi nimiž by se měl kruh vykreslit.
Pro vykreslení jednotlivých animací byly použity tyto návrhy:

2.8.1 Kružnice

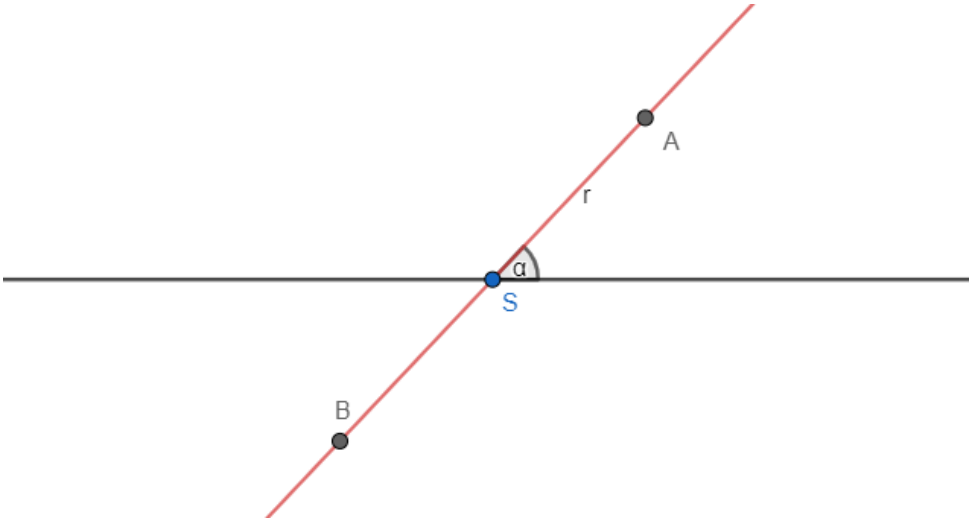
Animace kružnice je tvořena postupným vykreslováním bodu, který má od daného středu kružnice S konstantní vzdálenost r. Bod definující vykreslovanou kružnici o poloměru r je dán polárními souřadnicemi a má souřadnice $A = (s_1 + r \cdot cos(\alpha), s_2 + r \cdot sin(\alpha))$.



Obrázek 2.5: Návrh tvorby animace **kružnice**.

2.8.2 Přímka, úsečka

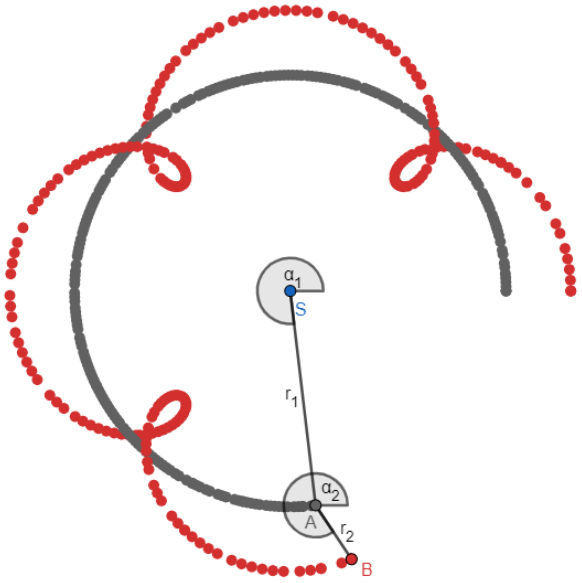
Pro vykreslení úsečky se použije stejný princip jako u vykreslování kružnice. Opět si určíme střed $S = (s_1, s_2)$. Dále použijeme dva body definované poloměrem r a úhlem natočení α. $A = (s_1 + r \cdot cos(\alpha), s_2 + r \cdot sin(\alpha))$, $B = -A$. Jejich spojnice pak definuje vykreslovanou přímku.



Obrázek 2.6: Návrh tvorby animace **přímky**.

2.8.3 Řetízek

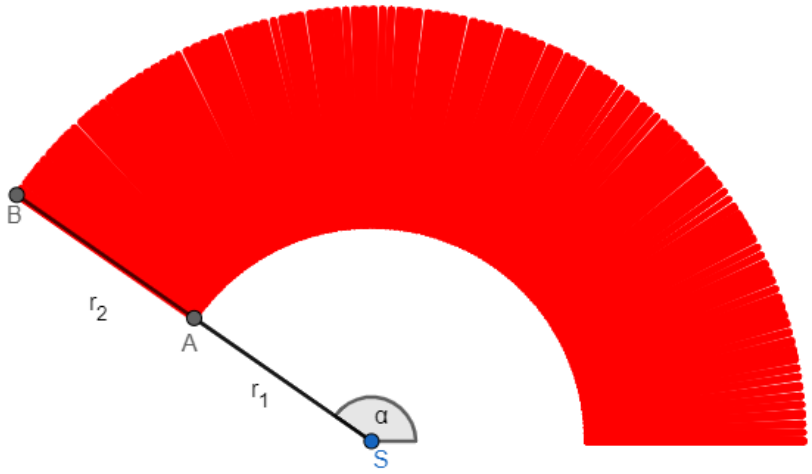
Animace řetízku vychází z animace kružnice. Daný postup ale zopakujeme dvakrát. Máme tedy střed $S = (s_1, s_2)$.
Dále máme bod $A = (s_1 + r_1 \cdot cos(\alpha_1), s_2 + r_1 \cdot sin(\alpha_1))$ definovaný poloměrem r_1 a úhlem natočení α_1. Nyní ale přidáme ještě jeden bod B definovaný středem A, poloměrem r_2 a úhlem natočení α_2.
$B = (s_1 + r_1 \cdot cos(\alpha_1) + r_2 \cdot cos(\alpha_2), s_2 + r_1 \cdot sin(\alpha_1) + r_2 \cdot sin(\alpha_2))$.



Obrázek 2.7: Návrh tvorby animace **řetízku**.

2.8.4 Kruh, mezikruží

Tvorba kruhu opět vychází z animace kružnice. Jedná se o vykreslení plochy útvaru. Mezikruží je tvořeno vnější a vnitřní kružnicí. Oba dva body definující kružnice mají stejný úhel natočení, ale jiný poloměr. Pro vyplnění oblasti mezi kružnicemi je vždy vykreslena úsečka spojující oba body definující kružnice. Jedná se tak o postupné vykreslování úseček do tvaru mezikruží.

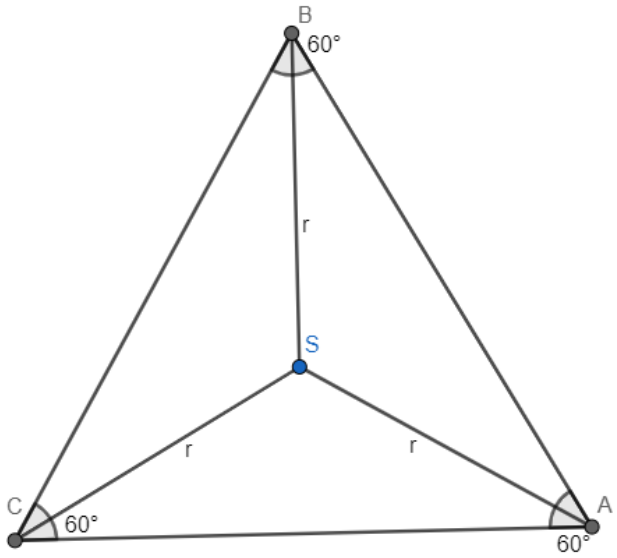


Obrázek 2.8: Návrh tvorby animace **mezikruží**.

Pokud bychom nastavili poloměr r_1 na nulu, vytvořili bychom tak kruh bez vnitřní prázdné části.

2.8.5 Trojúhelník

Pro vytvoření rovnostranného trojúhelníku je potřeba znát pouze střed a poloměr. Mezi každými dvěma vrcholy pak najdeme úhel o velikosti šedesát stupňů.



Obrázek 2.9: Návrh tvorby animace **trojúhelníku**.

Vrcholy trojúhelníku se středem $S = (s_1, s_2)$ mají následující souřadnice:
$A = (s_1 + r \cdot cos(\alpha), s_2 + r \cdot sin(\alpha))$
$B = (s_1 + r \cdot cos(\alpha + 60°), s_2 + r \cdot sin(\alpha + 60°))$
$C = (s_1 + r \cdot cos(\alpha + 120°), s_2 + r \cdot sin(\alpha + 120°))$
Kde r je vzdálenost vrcholu od středu a α je počáteční natočení vrcholu. Výsledný trojúhelník vznikne vykreslením úseček mezi jednotlivými vrcholy. Pro pohyb celým trojúhelníkem nám pak stačí měnit pouze souřadnice jeho středu a všechny vrcholy se budou pohybovat s ním.

Pokud bychom chtěli animovat jednotlivé vrcholy trojúhelníku beze změny souřadnic ostatních vrcholů, musíme u trojúhelníku definovat každý vrchol zvlášť.

Kapitola 3

Praktické řešení

V této kapitole je uveden postup implementace. Jmenovitě je zde popsaný výběr programovacího jazyka, výběr a testování použitých knihoven a také implementace komponent pro práci s kamerou a výslednou projekcí.

3.1 Výběr programovacího jazyka

Vzhledem k podstatě problému se zde hned od začátku nabízelo zvolit jako programovací jazyk **C++**. Pomocí tohoto jazyka se obecně píší programy tohoto typu, tedy programy zobrazující nějakou grafiku.
V tomto případě však byl zvolen jako programovací jazyk **Python**. Tento jazyk byl zvolen hned z několika důvodů:

- Jedná se o interpretovaný jazyk, tedy není potřeba před každým spuštěním kód kompilovat.
- Existují zde knihovny řešící problém rozpoznávání postav. Tyto knihovny jsou z části také psány v jazyku **C++**, ale není jich takové množství a nejsou tak modulární, aby mohly být použiti v tomto projektu.
- Výuka na FELu je zaměřená na programování v Pythonu. Předměty jako Umělá inteligence (UI) nebo Vidění robotů (VIR) se vyučuje v Pythonu. Z těchto předmětů byly brány základy tvorby programů.

Důvodů samozřejmě existuje mnohem více, ale hlavním impulsem byla jednoduchost testování programu při zkouškách. Bylo potřeba upravovat kód během zkoušek s tanečníky tak, aby animace správně reagovaly. Pokud by byl program psaný v jazyku **C++**, byl by určitě rychlejší a méně náročný na grafický výkon, nicméně výsledné animace nejsou nijak složité a celková rychlost je vysoká i v prostředí **Pythonu**, což bylo neustále testováno při implementaci.

3.2 Výběr knihoven

V tomto projektu byly vyhledány a použity knihovny, které co nejvíce zjednodušily práci s jednotlivými okny projekce a samotnou vykreslovanou grafikou. Při výběru knihoven byly zvažovány tyto a v tomto pořadí:

1. OpenCV [21]
2. Pygame [17]
3. Manim [13]
4. Taichi [18]
5. Arcade [16]

OpenCV:
Knihovna *OpenCV* [21] nabízí snadnou tvorbu grafiky v reálném čase a snadnou manipulaci s jednotlivými okny. Vykreslovaná grafika však není nijak propracovaná. Lze zde sice zobrazit všechny potřebné geometrické tvary, ale jejich vykreslení není vizuálně dostačující.

Pygame:
Knihovna *Pygame* [17] umožňuje velice jednoduchou správu oken. Nabízí oproti *OpenCV* o mnoho více možností v projekci jako jsou různé přechody animace a další její úpravy. Umožňuje také antialiasing. Vykreslovaná grafika však ale také není dostačující požadované kvalitě.

Manim:
Knihovna *Manim* [13] byla vytvořena pro animaci grafů a matematických zápisů. Vzhledem ke geometrické povaze projekce by její animace byla vyhovující, nicméně zatím tato knihovna neumožňuje vykreslovat projekci v reálném čase. Daná animace se vždy musí nejprve vytvořit a pak až se dá přehrávat.

Taichi:
Knihovna *Taichi* [18] byla dlouhou dobu považována za finální knihovnu. Dá se pomocí ní vytvářet jednoduchým způsobem základní animace, která svojí kvalitou odpovídá požadavkům na projekci. Knihovna umožňuje snadnou správu oken projekce. Jedinou nevýhodou a také důvodem, proč nakonec nebyla zvolena, je nedostatečná dokumentace. Většina z ní je totiž psaná v čínštině a jen její část je přeložena do angličtiny.

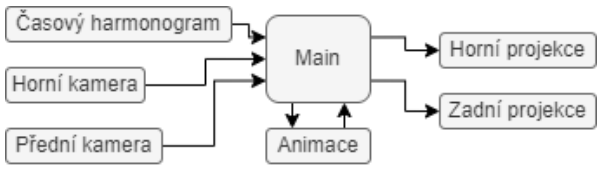
Arcade:
Finální a tedy i nakonec zvolenou knihovnou je knihovna *Arcade* [16]. Tato knihovna je vytvořena především pro tvorbu 2D her. Umožňuje snadnou správu oken projekce a jejich nastavení. Dá se s ní tvořit jednoduchá animace velice snadno a její vzhled je dostatečný. Podporuje také zobrazování objektů ve formě spritů[1]. Její hlavní výhodou je však její schopnost přehrávat animace vytvořené jako shadery[2](viz níže).

3.2.1 Celková struktura programu

Vzhledem k požadavku na jednoduchost, byl program navrhnut tak, aby bylo celkové uspořádání co možná nejintuitivnější a uživatelsky nejvíce přívětivé. Z diagramu 2.2 je vidět, že se celý program skládá z komponent, které řídí jednotlivé jeho části. Základním a také nejdůležitějším prvkem je program ovládající a spravující vykreslování grafiky. Obsahuje komponenty starající se o obě okna projekce, data z kamer a také vstupní informace o časovém harmonogramu představení. Má také přístup do složek obsahujících jednotlivé soubory animací a grafiky.

[1]Sprite je označení používané v počítačové grafice pro většinou malý dvourozměrný obrázek, který je integrován do větší scény.

[2]Shader je počítačový program sloužící k řízení jednotlivých částí programovatelného grafického řetězce grafické karty.



Obrázek 3.1: Struktura řídícího programu

3.2.2 Řízení představení v čase

Při implementaci bylo nutné vytvořit systém, který by se staral o spouštění jednotlivých animací a dalších komponent podle předem navržené choreografie. Jako univerzální formát byl zvolen textový soubor, který na každém řádku obsahuje informaci o jedné z vykreslovaných animací nebo komponentě.
Pro srozumitelnost byla navržena tato notace, podle které byl vytvořen celý harmonogram představení.
Každý řádek obsahuje šest částí oddělených mezerou, které se čtou zleva doprava. Jednotlivé části znamenají:

- 1. část: určuje, zdali se jedná o projekci na zem(H) nebo projekci na zadní plátno(P)
- 2. část: určuje, ve který čas má daný efekt končit (například 10. sekunda)
- 3. část: definuje název spouštěného efektu i se vstupními inicializačními parametry (například *Circle(x,y)*)
- 4. část: definuje parametry používané při aktualizaci efektu (například *(time)*)
- 5. část: definuje parametry používané při vykreslování efektu (například *(dx,dy)*)
- 6. část: říká, zdali je potřeba spouštět detekci lidského těla (*True/False*)

Pokud řádek v souboru na začátku neobsahuje ani *H* ani *P*, řádek se při čtení přeskočí. Tímto způsobem se v dokumentu dají vytvářet komentáře. Jednoduchý dokument by mohl vypadat například takto:

```
1 #----------------------Uvod----------------------
2 #---------------Projekce na zadni platno----------
3 P 10 Circle(x,y) (time) () True
4 P 100 Black() () () False
5 #--------------Projekce na zem--------------------
6 H 15 Line(x1,y1,x2,y2) () (dx,dy) False
7 H 100 Black() () () False
```

Listing 3.1: Příklad textového dokumentu.

Na třetím řádku je tedy řečeno, že se animace bude promítat na zadní plátno, že se jedná o animaci s názvem Circle, která by měla končit v čase 10 sekund a že je při ní potřeba spustit detekci postavy tanečníka. Z tohoto příkladu je vidět, že tam, kde jsou prázdné závorky, nejsou potřeba žádné argumenty.

3.3 Detekce

Detekce v programu probíhají dvě, a to detekce pomocí přední kamery, která má za cíl detekovat body na lidském těle, a pak také detekce pomocí horní kamery, která detekuje pouze přítomnost člověka ve vymezeném prostoru jeviště. Dále zde je ještě komponenta detekující gesta tanečníka využívající body detekované pomocí přední kamery.

3.3.1 Detekce pomocí přední kamery

Tato detekce využívá balíček *openVINO*[22].
Tento balíček je již naučená neuronová síť pro detekci bodů na lidském těle. Balíček si bere obraz z kamery pomocí knihovny *OpenCV* [21], vždy každý snímek projde neuronovou sítí a výstupem je množina bodů detekovaných neuronovou sítí.
Výstupem je pole o velikosti *k*l*m*, kde $k = 10$ udává číslo detekovaného člověka (maximum 10 lidí najednou), $l = 17$ udává daný bod na lidském těle a $m = 2$ udává souřadnice daného bodu v ose x a y.
Příklad výstupních dat z této funkce je následující:

```
pose = [
[[1.23, 2.45],[4.85, 9.56],...,[3.12, 1.12],[5.45, 6.21]],
[[4.21, 6.21],[0,0],[0,0],..,[2.11, 3.23],[5.12, 1.12]],
[[0,0],[0,0],[0,0],[0,0],[0,0],...,[0,0],[0,0],[0,0]],
...
[[0,0],[0,0],[0,0],[0,0],[0,0],...,[0,0],[0,0],[0,0]]]
```

Z tohoto příkladu je vidět, že byly detekovány 2 osoby, přičemž u druhé osoby nebyly detekovány všechny body na těle. To může být způsobeno například faktem, že se daná část těla nachází v zákrytu jiné, nebo se postava dostala do záběru kamery ne úplně celá. Algoritmus sice dokáže chybějící body předpovědět, ale není vždy stoprocentně správný.

3.3.2 Detekce pomocí horní kamery

Výstupem této detekce je pouze souřadnice středu těla člověka pohybujícího se v dané části jeviště. Detekce využívá opět knihovnu *OpenCV* [21], pomocí které každý snímek z kamery nejprve rozostří, převede na černobílý, detekuje kontury u tmavých částí a vrátí souřadnice jejich středu.
Výstupem z této funkce je pole velikosti $m*n$, kde m udává počet detekovaných postav a $n = 2$ udává polohu dané postavy v ose x a y.
Příklad výstupních dat z této funkce je následující:

```
1 pose = [[1.21, 3.12], [5.21, 1.12], [3.23, 4.11]]
```

Z tohoto příkladu je vidět, že se detekovaly celkem tři postavy.

3.3.3 Detekce gest tanečníka

Při detekci jednotlivých gest se využívá dat získaných z detekce pomocí přední kamery. Polohy jednotlivých kloubů lidského těla se porovnávají s předem vytvořenými podmínkami pro jednotlivé gesto. Jak již bylo řečeno v teoretické části, gest je zde vytvořeno více pro případ, že jedno nebo více z nich nemůžeme z nějakého důvodu použít.
Podmínky pro definici jednotlivého gesta vycházejí z polohy kloubů tanečníka vykonávající dané gesto. Poloha kloubů je orientační a závislá vždy na poloze ostatních jeho kloubů, ne na poloze v prostoru jeviště. Porovnáváme tedy vždy každý kloub těla s jiným. U každého kloubu koukáme na obě jeho souřadnice a kontrolujeme, jestli se nachází na správném místě.
Pokud máme například gesto, které tvoří člověk stojící snožmo s rukama podél těla, kontrolujeme například, že se klouby zápěstí nacházejí pod lokty a nad koleny. Pokud použijeme dostatečně mnoho podmínek, je možné od sebe jednotlivá gesta dobře rozlišit.

3.3.4 Projekce

Projekční program se stará o správnou projekci animací na vybrané místo. Struktura pro oboje projekce je shodná. Můžeme si jí vysvětlit na následujícím shrnutí:

```
1  class Projekce:
2      def init(width, height, dt):
3          detection
4          line_from_file
5          dt -> time
6      def update_from_file(time):
7          get line_from_file -> animation
8      def on_update(dt):
9          update animation
10         update detection
11         time + dt
12     def on_draw():
13         draw animation
```

Listing 3.2: Shrnutí struktury projekce

Z tohoto shrnutí je vidět, že v prvním řádku se deklaruje projekce. Ve druhém řádku se pak inicializuje okno projekce s danou šířkou a výškou. Dále se pak v této inicializační funkci spustí komponent pro detekci tanečníků pomocí dat z kamery, nastaví se první řádek z časového harmonogramu a upraví se počáteční čas pro spuštění. Čas spuštění se dá nastavit při spuštění celého programu. Je to důležité hlavně ve fázi testování, kdy se nemusí spouštět celé představení vždy od začátku, ale může se nastavit počáteční čas od kterého chceme začít vykreslovat.
Na řádku šest je pak definice funkce aktualizující informace z časového harmonogramu. Tato funkce vždy najde další řádek v textovém souboru příslušný k dané projekci.
Na řádku osm pak vidíme funkci, která aktualizuje čas, probíhající animaci a také detekci tanečníků.
Na dvanáctém řádku je funkce vykreslující danou animaci.

3.4 Animace

Animace použité v tomto projektu jsou, jak již bylo řečeno v teoretickém úvodu, rozděleny do dvou kategorií. První z nich jsou animace vytvořeny přímo pomocí knihovny *Arcade* [16], která umožňuje pracovat jak ze základními grafickými objekty jako je například kruh a čtverec, tak také umožňuje vykreslování obrázků jako sprity. Druhá kategorie jsou animace vytvořené jako shadery v jazyku GLSL [19].

Základní animace vytvořené pomocí knihovny Arcade

Jak již bylo zmíněno, knihovna Arcade je vytvořena primárně pro tvorbu 2D her. Obsahuje tedy i řadu možností, jak vytvořit a animovat geometrické objekty. Knihovna obsahuje možnost tvorby těchto geometrických objektů:

- oblouk (část kruhu)
- kruh
- elipsa
- přímka
- obdélník
- parabola
- bod
- polygon
- trojúhelník

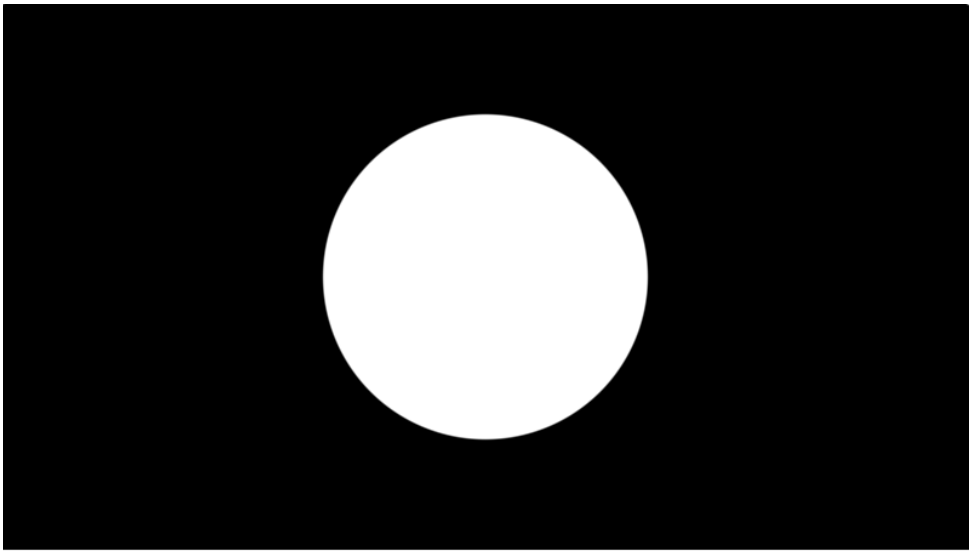
Mimo tyto tvary také umožňuje vykreslit jakýkoli "obrázek" jako sprite. Vykreslování spritů je jednoduchý proces, při kterém se načtený obrázek vždy upraví podle zadaných parametrů a umístí se na zadané místo. Tento způsob je velice jednoduchý a výpočetně nenáročný. Navíc se u takto vytvořených tvarů dají detekovat kolize s ostatními sprity.

GLSL Shadery

GLSL (OpenGL Shading Language) je programovací jazyk podobný **C**, který je vytvořený výhradně pro programování grafiky.
Animace psané pomocí GLSL shaderů nejsou výpočetně náročné a jsou vizuálně nepřekonatelné.
Velikou výhodou je, že zápis není nijak dlouhý a i pomocí pár řádek kódu se dá vytvořit opravdu působivá animace.

```
1 void mainImage( out vec4 fragColor, in vec2 fragCoord ){
2     vec2 uv = ( fragCoord -.5* iResolution.xy)/iResolution.y;
3     float d = length(uv);
4     float r = 0.3;
5     float color =  smoothstep(r,r - 2.0/iResolution.y ,d) ;
6     fragColor = vec4(vec3(color),1.0);
7 }
```

Listing 3.3: Příklad zápisu GLSL shaderu.



Obrázek 3.2: Příklad GLSL shaderu.

Z tohoto příkladu je vidět, že GLSL shadery pro zápis vycházejí primárně z matematického popisu systému. Zápis každé animace tedy musí být dobře matematicky popsatelný. Pokud tomu tak je, půjde animace velice jednoduše vytvořit.
Pro animování je zde použita proměnná času **iTime**, která roste s přibývajícím časem. Pomocí ní se tvoří veškeré animované přechody.

3.5 Optimalizace

Tím, že byl zvolen jako programovací jazyk Python, se zvýšil také předpoklad vyššího výpočetního výkonu. Chceme-li ale spouštět animace i na ne příliš výkonném počítači, musíme se snažit napsat program co možná nejrychlejší. Program se dá urychlit správnou volbou vykreslování animací a také správným nastavením celého programu.

3.5.1 Rychlost vykreslování animace

Při implementaci se už od začátku dbalo na co nejmenší výpočetní náročnost programu. Vzhledem k jednotlivým animacím, které jsou někdy docela složité, bylo testováno mnoho variant vykreslování. Pokud při animaci nebyla potřeba data z kamery a celková animace byla jednodušší na animování (byl zde například jen jeden pohybující se objekt), byl zvolen způsob vykreslování pomocí funkcí GLSL shaderů [19].
Pokud ale objektů na vykreslování bylo mnohem více, nebo bylo potřeba reagovat na pohyb tanečníků, byly animace nejprve tvořeny pomocí základních geometrických funkcí v knihovně Arcade [16].
Takto vytvořené animace s rostoucím počtem vykreslovaných objektů začaly být stále výpočetně náročnější. Proto byla nejprve zvolena varianta, která všechny vykreslované objekty zabalí do jednoho seznamu, který pak celý najednou vykresluje. To pomohlo v rychlosti vykreslování, ale nebylo to zdaleka nejoptimálnější řešení.
Nakonec byla zvolena varianta vykreslovat dané geometrické objekty pomocí předem připravených obrázků jako sprity. Takto vytvořené animace nebyly zdaleka tak výpočetně náročné, protože se vždy nemusel každý daný geometrický objekt generovat zvlášť, ale vždy se načetl jen jeden obrázek na začátku funkce a pak se jen vykresloval znovu a znovu na jiná místa.
Jako příklad můžeme uvést animaci s názvem *MiniCircles*, která má za cíl vykreslovat zvětšující se kružnice za tanečníkem tam, kudy prochází. V průběhu dané animace se tak vykresluje více než 500 kružnic. Tyto kružnice se pouze zvětšují do dané maximální velikosti. Vždy při každé aktualizaci polohy tanečníků se tedy pod tanečníkem vytvoří nová kružnice a v každé další aktualizace se její poloměr zvětšuje.
Pokud srovnáme počet snímků za sekundu při vykreslování pomocí funkcí knihovny Arcade a pomocí spritů, rozdíl se blíží až ke dvaceti FPS ve prospěch spritů.

3.5.2 Celkové zrychlení programu

V průběhu vystoupení je potřeba detekovat polohu tanečníků. Poloha není ale potřebná detekovat neustále. Animace využívající detekovanou polohu se ve vystoupení vyskytují v menší míře oproti ostatním animacím. Dále jsou ve vystoupení také pauzy bez animace. Pro tyto případy byla do programu implementována funkce spustit detekci jen v okamžiku potřeby. Detekce se dá spouštět v textovém souboru s časovou osou (viz příklad textového dokumentu 3.1) v posledním argumentu. *True/False* určuje, že je detekce zapnutá či ne. Dalšího zrychlení se docílilo implementací prázdné funkce. Tato funkce vypadá následovně:

```
1 class Black:
2     def __init__(self):
3         self.time = 0
4     def update(self):
5         pass
6     def draw(self):
7         pass
```

Listing 3.4: Prázdná funkce

Tato prázdná funkce je spouštěna vždy, když se dokončí požadovaná animace a ještě není čas spouštět žádnou další animaci. To zaručuje, že detekce tanečníků je prováděna opravdu jen nezbytně krátkou dobu.

3.5.3 Korekce času animací

Během vystoupení se někdy stane, že se čas spuštění animací posune. To je zapříčiněno spouštěním detekce postav, nebo vykreslením složitější animace. Během testování se posun animací někdy přiblížil i ke dvěma sekundám. Takto velký časový rozdíl je hodně patrný u animací přesně sladěných s hudbou a taneční choreografií.
Pro minimalizaci tohoto časového posunu byla implementována funkce umožňující při běhu vystoupení měnit čas projekce. Pomocí dvou kláves je možné přičíst nebo odečíst vždy desetinu sekundy.
Tato funkcionalita byla otestována během vystoupení v divadle, kde byla použita ke korekci času vícekrát během celého vystoupení.

3.5.4 Korekce snímaného prostoru

Při snímání prostoru přední i horní kamerou dochází ke zkreslení účinkem perspektivy. Pro odstranění tohoto problému byly implementovány dvě podobné komponenty každá pro jednu kameru, které ze vstupních dat z kamery vyříznou určitou část definovanou uživatelem, která pak odstraní perspektivu. Komponenta funguje tak, že po jejím spuštění uživatel vybere čtyřmi body část snímané plochy tak, aby zabírala přesně ten prostor, který je potřeba. Komponenta následně vybraná data uloží do souboru, který se vždy při inicializaci snímání kamerou načte a kamera se podle těchto dat ořízne.

Obrázek 3.3: Výstup z přední kamery.

Obrázek 3.4: Oříznutý výstup z komponenty korekce kamery.

Kapitola 4

Jednotlivé animace a jejich podrobný rozbor

V této kapitole jsou sepsány všechny efekty použité ve vystoupení. Každý efekt je popsán zvlášť a je u něj vysvětlena jeho funkce a postup implementace. U každé animace je obrázek její projekce a u některých i snímek z živého vystoupení. Snímky z vystoupení byly barevně poupraveny, takže se zde objevují i jiné barvy projekce, která je ve skutečnosti bílá.
Fotky z vystoupení byly pořízeny *Andreou Jircovou* a *Michalem Hančovským*.

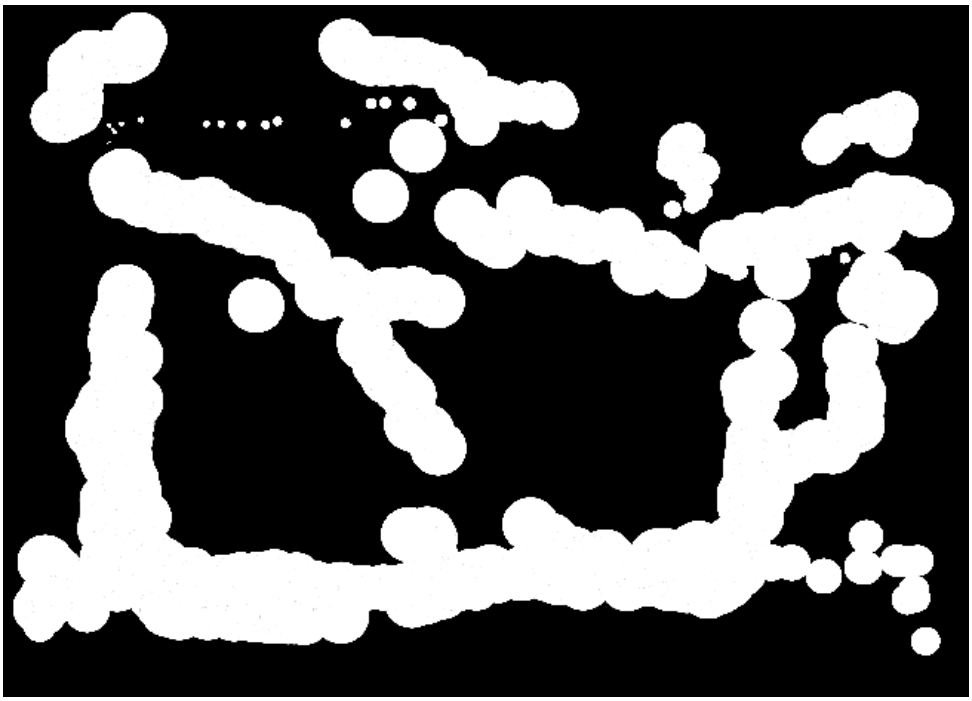
4.1 Animace vytvořené pomocí knihovny Arcade:

Všechny animace vytvořené pomocí knihovny *Arcade* byly implementovány celé. Nic nebylo odnikud převzato. Jednotlivé animace byly tvořeny přesně pro zadanou choreografii.

4.1.1 Kruhy

Tato animace je, jak již bylo popsáno výše, animací vykreslující kruhy v místě, kudy projde tanečník. Jsou zde tedy potřeba data z horní kamery.
Implementace:
Na detekovaných místech jsou vykresleny kruhy nulové velikosti a ty jsou přidány do celkového seznamu kruhů. V další detekci se pak všechny kruhy v

seznamu zvětší o danou velikost až do maxima. Kruhy zůstávají na taneční ploše až do konce celé animace.

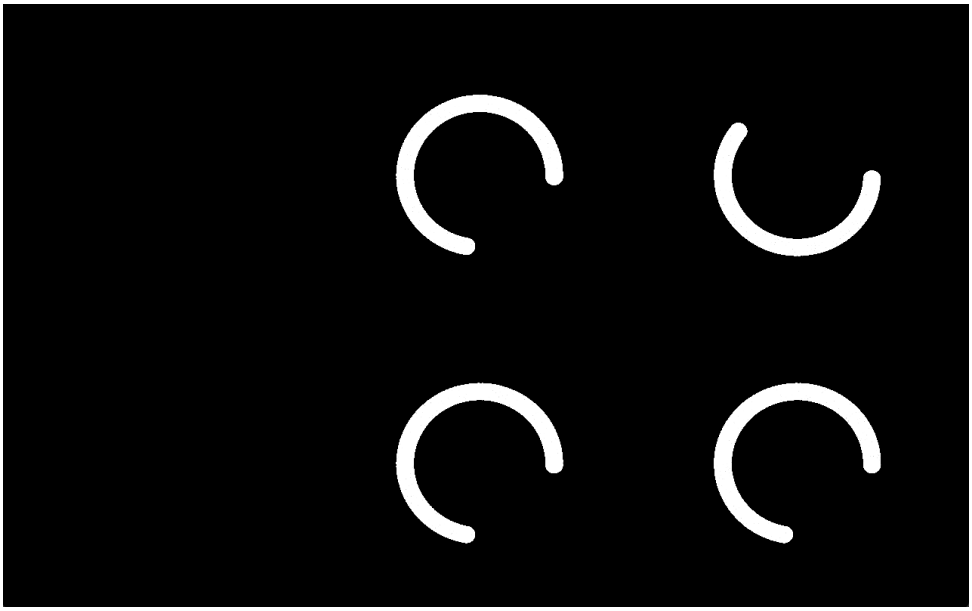


Obrázek 4.1: Vykreslená animace **Kruhy**.

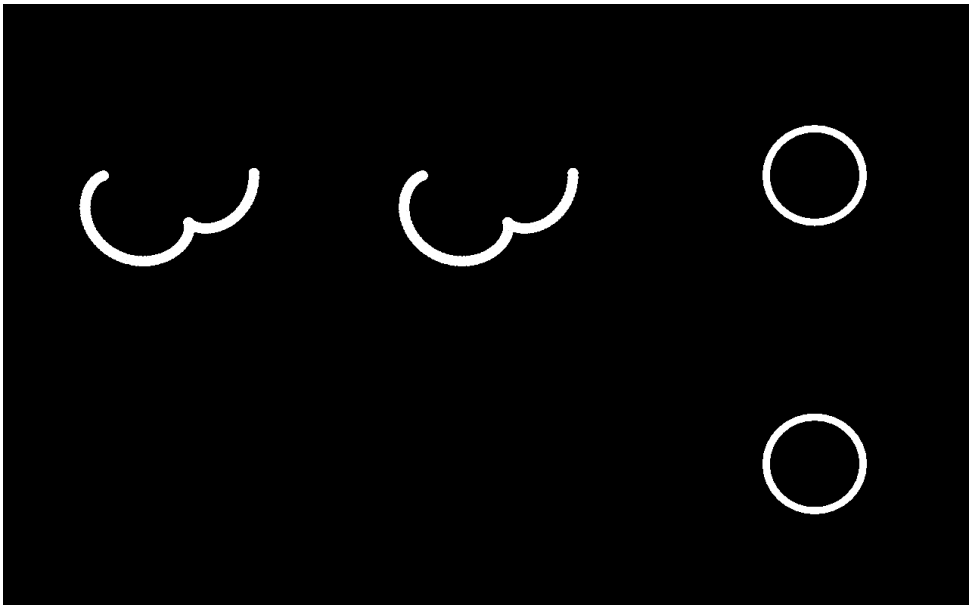
Obrázek 4.2: Animace **Kruhy** na vystoupení.

4.1.2 Řetízky

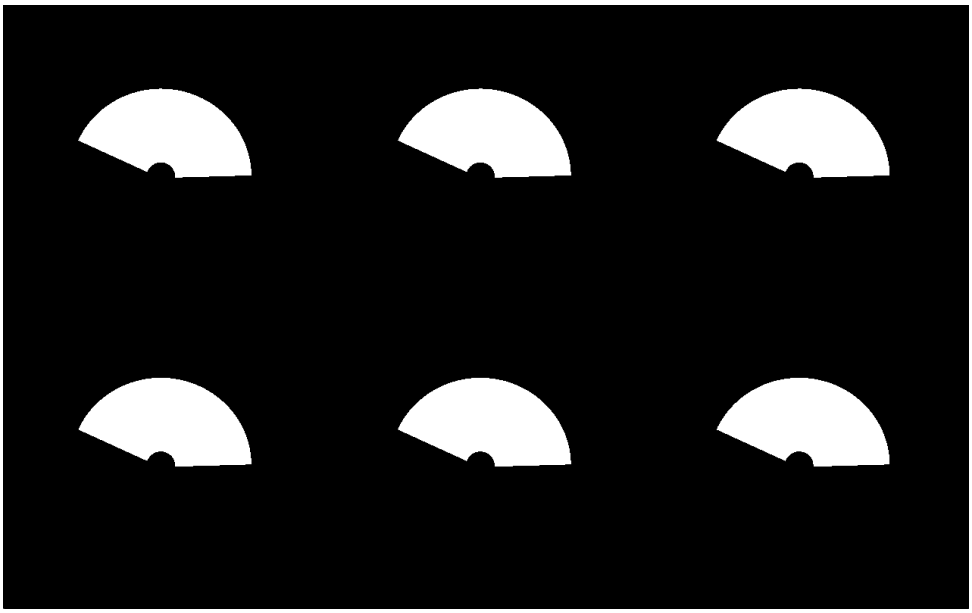
Animace s tímto názvem vznikla spojením čtyř animací v jednu. Animace se vykresluje na zem pod tanečníky bez potřeby znát jejich polohu. Jedná se o postupné vykreslování kružnic kolem tanečníků a jejich následné mazání, vykreslování "řetízku" (spirály) kolem tanečníků, pohyb kružnice do strany od tanečníka a zpět a nakonec vykreslení kruhu pomocí kruhových výsečí.
Animace byly spojeny do jedné z důvodu přesné časové následnosti, která odpovídá taneční choreografii.
Implementace:
V první části se vykreslují kružnice kolem tanečníků. Kružnice jsou vykreslovány způsobem popsaným v teoretické části (obrázek 2.5).
V druhé části se vykreslují řetízky opět podle postupu popsaného výše (obrázek 2.7).
Třetí část je obyčejné vykreslení kruhu s poloměrem. Poloha středu kruhu se pak pohybuje podle potřeby.
Ve čtvrté části je vykreslován kruh popsaný opět výše v teoretické části (obrázek 2.8).



Obrázek 4.3: Vykreslená animace **Řetízky**(1. část s kolečky).



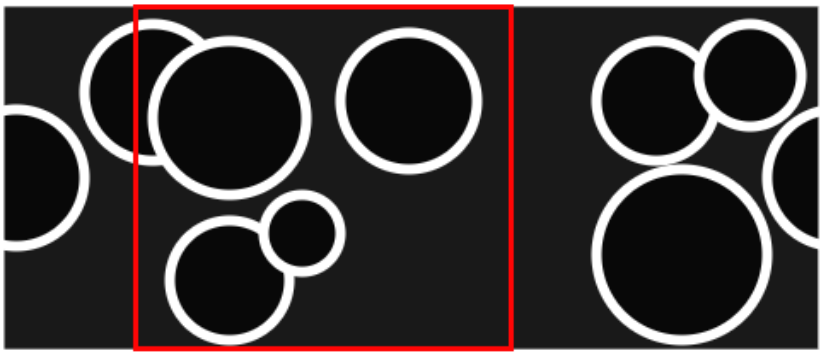
Obrázek 4.4: Vykreslená animace **Řetízky**(2. a 3. část s řetízky a kolečky).



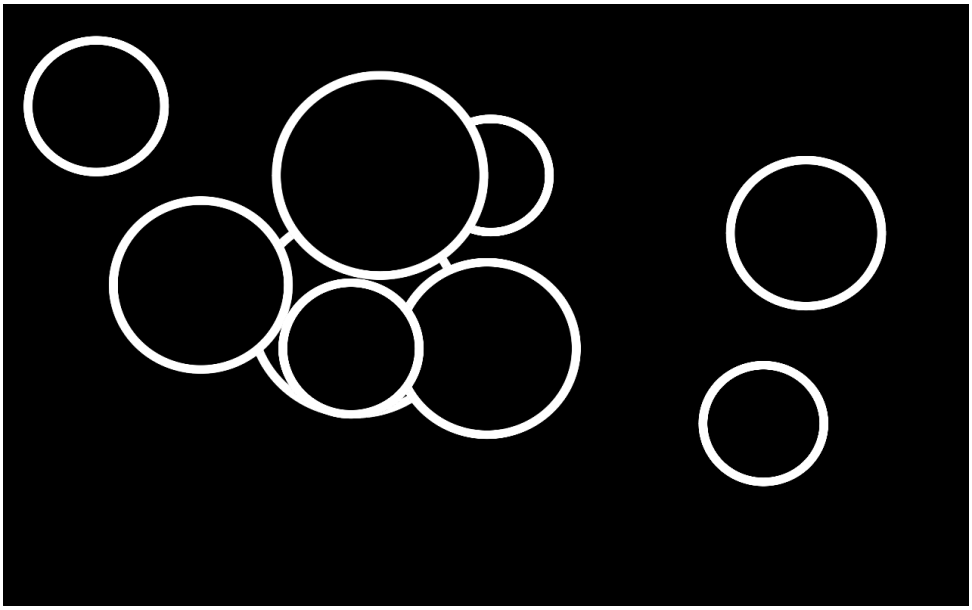
Obrázek 4.5: Vykreslená animace **Řetízky**(4. část s kruhy).

4.1.3 Ostrůvky

Vznik této animace je spojen s improvizací tanečníků. Kruhy neboli ostrůvky se nejprve objevují v předem definovaných místech. Poté se všechny postupně posunují vpravo. Tím vznikají jakási "souostroví"po kterých se tanečníci pohybují. Na konci animace pak jednotlivé ostrůvky opět mizí a tím zaniká i tanečníkův pohyb.
Implementace:
Při implementaci byly určeny body, na kterých se mají ostrůvky objevovat. Celkem se ve viditelné části jeviště objeví šest ostrůvků. Celková plocha animace je však velikostně dvojnásobná. V neviděné části se také vytvoří 6 ostrůvků. Všechny se pak v daném čase začnou pohybovat vpravo. Pokud se některý z nich dostane na kraj animace, přechází plynule na druhý kraj vlevo. Tím, že byl vytvořen dvojnásobný počet ostrůvků, nedochází k tomu, že by divák viděl, jak ostrůvek na jedné straně mizí a na druhé se opět objevuje. Je tak i pro tanečníky náhoda, kde se další ostrůvek objeví. Během vystoupení tak mají možnost improvizace.
Velikosti vykreslovaných ostrůvků jsou také náhodné. Je zde určeno rozmezí, aby se nestalo, že bude ostrůvek buď moc malý, nebo naopak zakryje celou plochu.
Celkový návrh je vidět na následujícím obrázku, kde červenou barvou je znázorněna oblast, která se promítá na jeviště.



Obrázek 4.6: Návrh animace **Ostrůvky**.

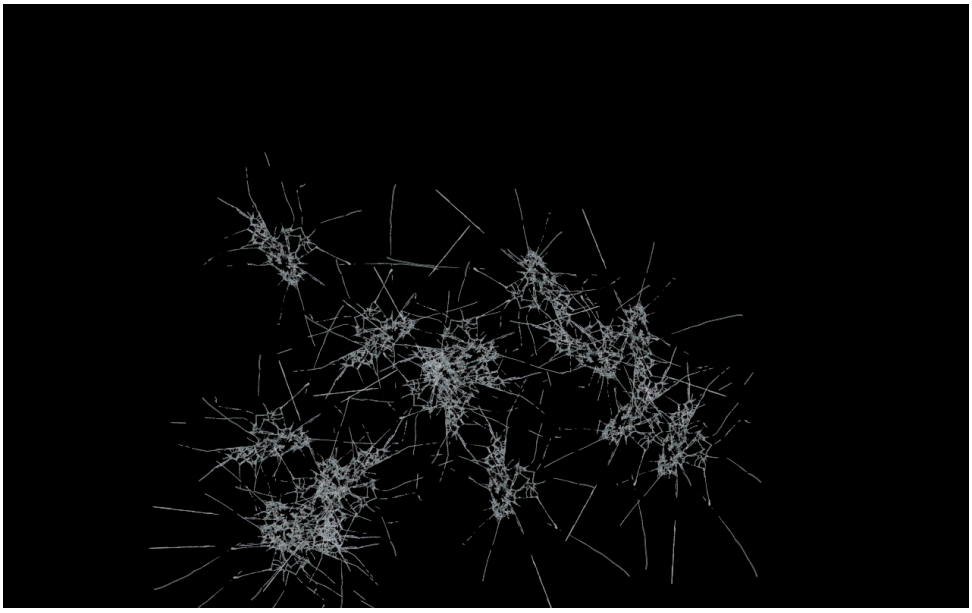


Obrázek 4.7: Vykreslená animace **Ostrůvky**.

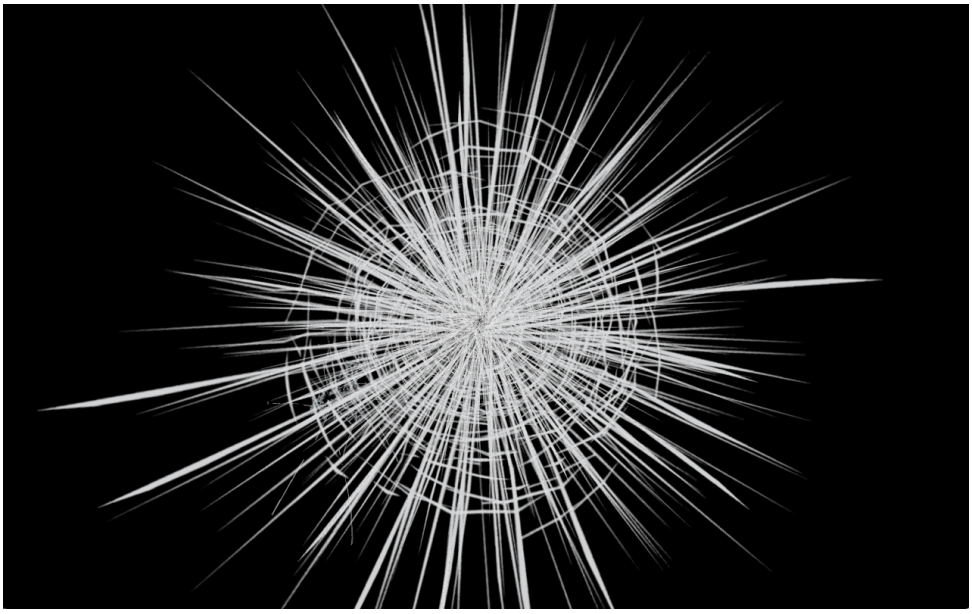
Obrázek 4.8: Animace **Ostrůvky** na vystoupení.

4.1.4 Prasklina

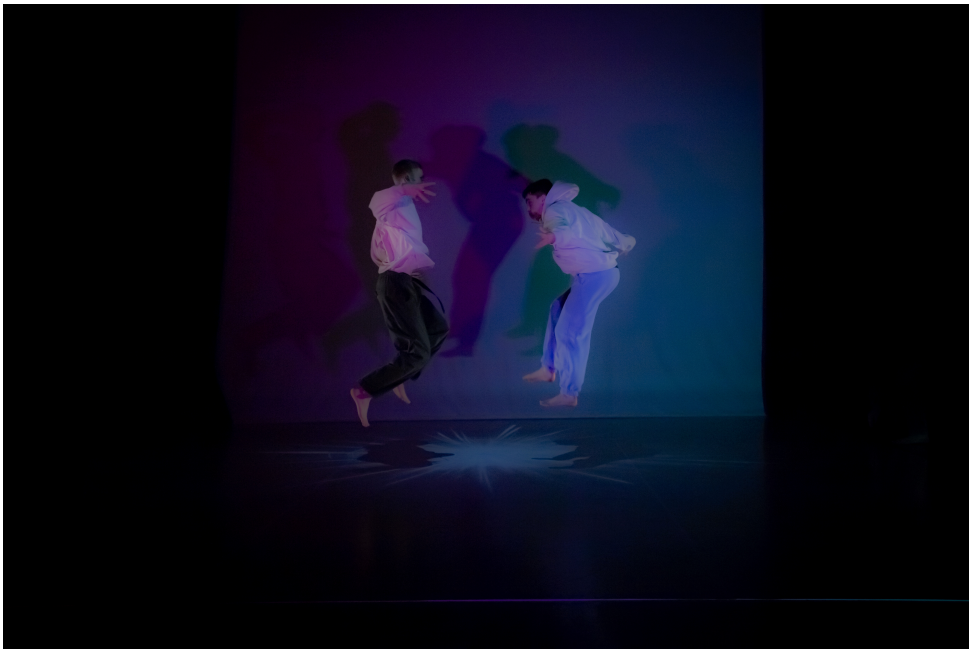
Pavučina nebo prasklina byla hlavní myšlenka při tvorbě této animace. Nejprve se během deseti sekund objevují drobné praskliny pod nohama tanečníků. Následně se od středu jeviště začíná objevovat čím dál větší prasklina zabírající nakonec celé jeviště. Po určité době se pak prasklina opět zmenší až do jejího úplného zániku.
Implementace:
Vykreslované tvary prasklin jsou uloženy jako obrázek, který se vždy vykreslí na místo určené detekcí polohy tanečníka. Aby nebyly praskliny všechny stejné, mění se jejich pootočení. Stejně tak to platí i pro velkou prasklinu uprostřed, která se jen vykresluje vždy ve středu jeviště.



Obrázek 4.9: Vykreslená animace **Prasklina** (praskliny pod tanečníky).



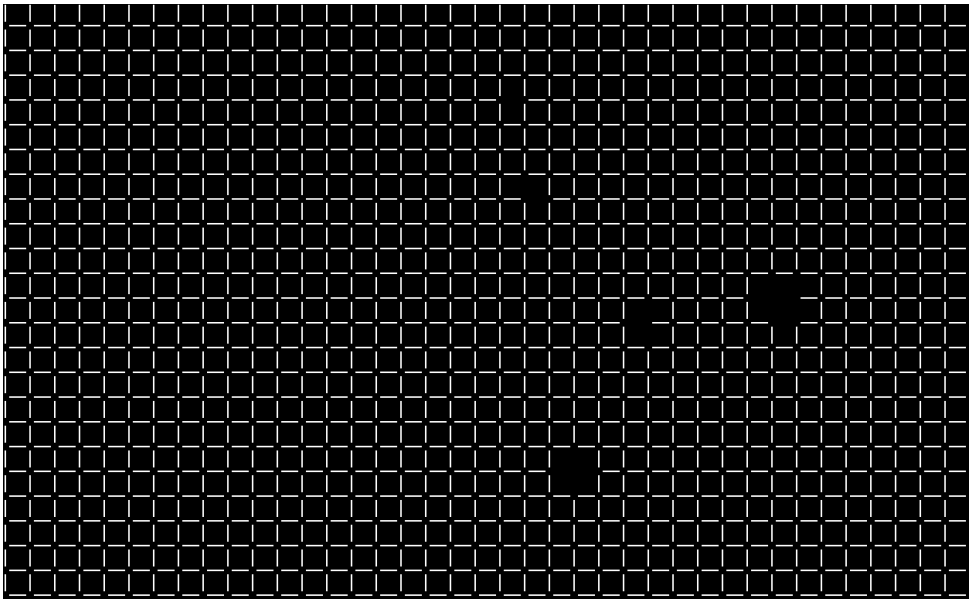
Obrázek 4.10: Vykreslená animace **Prasklina** (velká prasklina).



Obrázek 4.11: Animace **Prasklina** na vystoupení.

4.1.5 Síť

Cílem této animace bylo vytvořit efekt "prožírání sítě", neboli nějakým způsobem mazat vytvořenou mřížku při průchodu tanečníka. Animace reaguje na polohu tanečníkových končetin, přesněji řečeno na jeho zápěstí. Pohybem ruky tedy tanečník umazává jednotlivé části mřížky.
Implementace:
Síť je tvořena horizontálně a vertikálně umístěnými úsečkami. Ty jsou umísťovány postupně s předem definovaným rozestupem, aby pokryly celou plochu. Při mazání tanečníkem se v místech tanečníkových zápěstí vytvoří kruhy se zadaným poloměrem, které nejsou vykreslovány, ale které při kolizi s přímkou tuto přímku smažou. Tvoří se tak efekt "prožírání sítě".

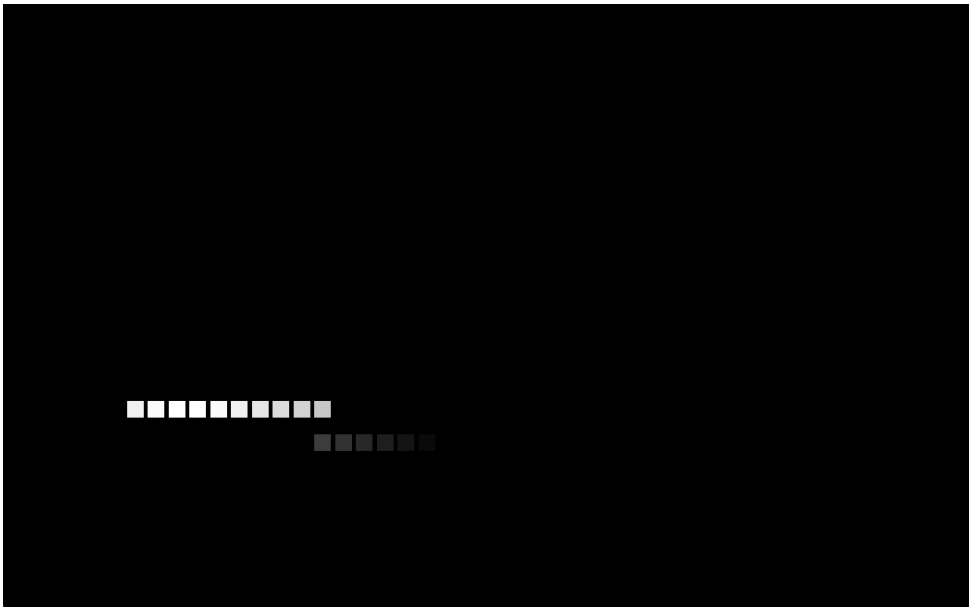


Obrázek 4.12: Vykreslená animace **Síť**.

Obrázek 4.13: Animace **Síť** na vystoupení.

4.1.6 Obdélník

Tato animace má znázorňovat přesun obdélníku za tanečníkem. Obdélník se posunuje přesně podle tanečníkových pohybů. Celá animace se skládá ze tří částí. V první se objeví samotný obdélník a pohybuje se spolu s tanečníkem. V druhé části se vytváří linky z malých obdélníků vždy ve směru ruky tanečníka. Ve třetí části se opět ukáže prvotní obdélník a pohybuje se spolu s tanečníkem.
Implementace:
Obdélník je načten jako obrázek, který akorát v průběhu animace mění polohu a natočení.
Linky z obdélníků jsou tvořeny pomocí stejného obrázku, který je jen vykreslován za sebou vždy desetkrát v daném směru, přičemž jeho intenzita se zvětšuje v průběhu vykreslování.

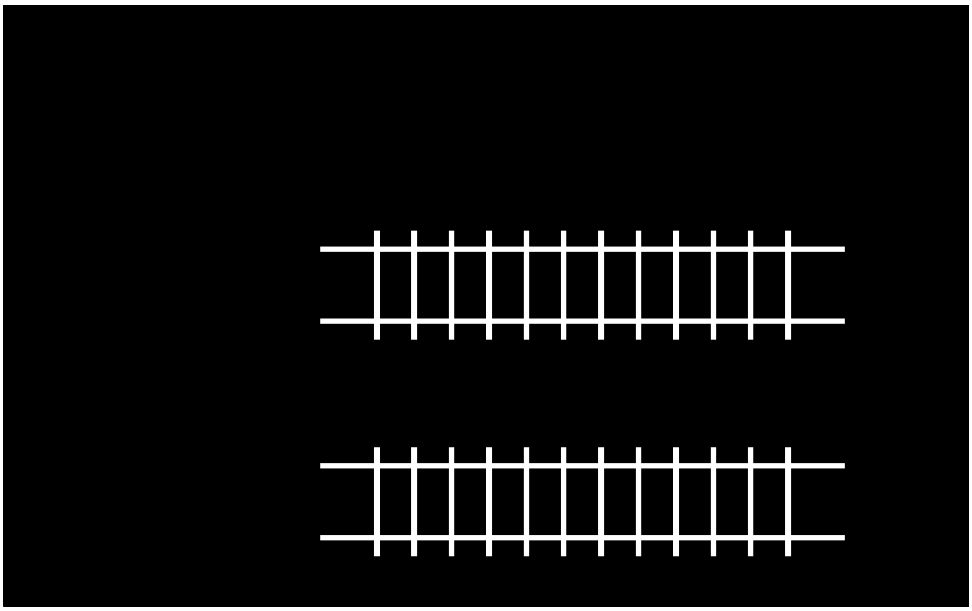


Obrázek 4.14: Vykreslená animace **Obdélník** (druhá část).

Obrázek 4.15: Animace **Obdélník** na vystoupení (první část).

4.1.7 Žebříky

V této animaci jde jen o vykreslení dvou žebříků na zem pod tanečníky, kteří se po nich pohybují z jedné strany jeviště na druhou. Při pohybu se žebříky objevují a následně také mizí ve směru pohybu tanečníků.
Implementace:
Žebříky jsou tvořeny přímkami. Ty jsou vždy tři pro každý stupeň žebříku. Dvě z nich tvoří strany a třetí příčku. Při vykreslování tedy dochází k plynulému přechodu.



Obrázek 4.16: Vykreslená animace **Žebříky**.

4.1.8 Pruhy

Při tvorbě této animace bylo vycházeno z požadavku na souboj tanečníků. Choreograficky tato animace má znázorňovat předbíhání tanečníků mezi sebou v jakémsi závodě. Animaci tvoří tři pruhy vždy od zadní části jeviště dopředu. Pruhy reagují na polohu tanečníků a zvětšují se, popřípadě zmenšují, podle jejich pohybu po jevišti. Ke konci animace zůstává pouze jeden pruh, který následně také mizí.
V animaci je také implementován blikající efekt, který náhodně rozsvítí a hned zhasne celou taneční plochu. Tento efekt má podtrhnout napjatou atmosféru závodu.
Implementace:
Každý z pruhů je tvořen přímkou určenou dvěma body. Jedním z nich je

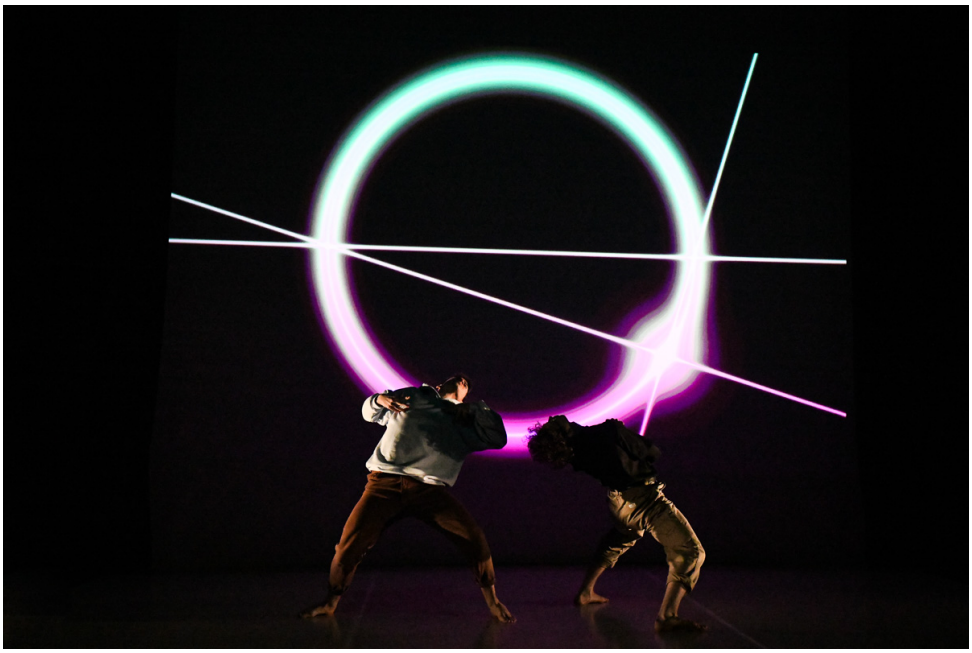
předdefinovaný bod v zadní části jeviště. Druhým bodem je tanečník. Aby se pruhy pohybovaly jen v y-ové ose, souřadnice druhého bodu má vždy x-ovou souřadnici stejnou a y-ovou si bere od polohy tanečníka.
Blikající efekt je tvořen vykreslováním obdélníku přes celou plochu, který mění svojí velikost náhodně z nuly na sto procent.

Obrázek 4.17: Vykreslená animace **Pruhy**.

Obrázek 4.18: Animace **Pruhy** na vystoupení.

4.2 Animace vytvořené pomocí GLSL shaderů

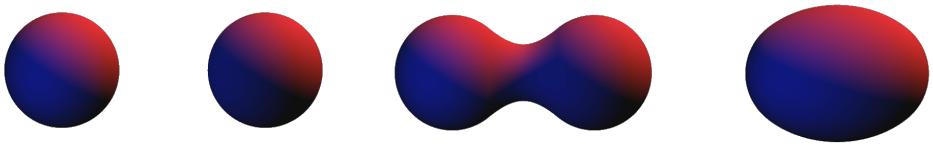
Všechny animace popsané níže jsou vytvořené pomocí jazyka GLSL a jsou předem připravené. To znamená, že nereagují nijak na tanečníky, ani se samy nijak náhodně nemění. Animace jsou většinou přesně definované, jednak aby zapadaly do hudby, a také proto, že jsou složitější a mají svůj výrazový význam.
Při implementaci byla brána inspirace z databáze animací na stránce *www.shadertoy.com* [23], pomocí níž byly také všechny animace implementovány a testovány. Žádná z animací ale není převzatá. Pokud u animací bylo něco převzato z dostupné databáze, bylo vše poupraveno tak, aby výsledná animace splňovala potřeby dané choreografií a vizuálně zapadala co celého celku.



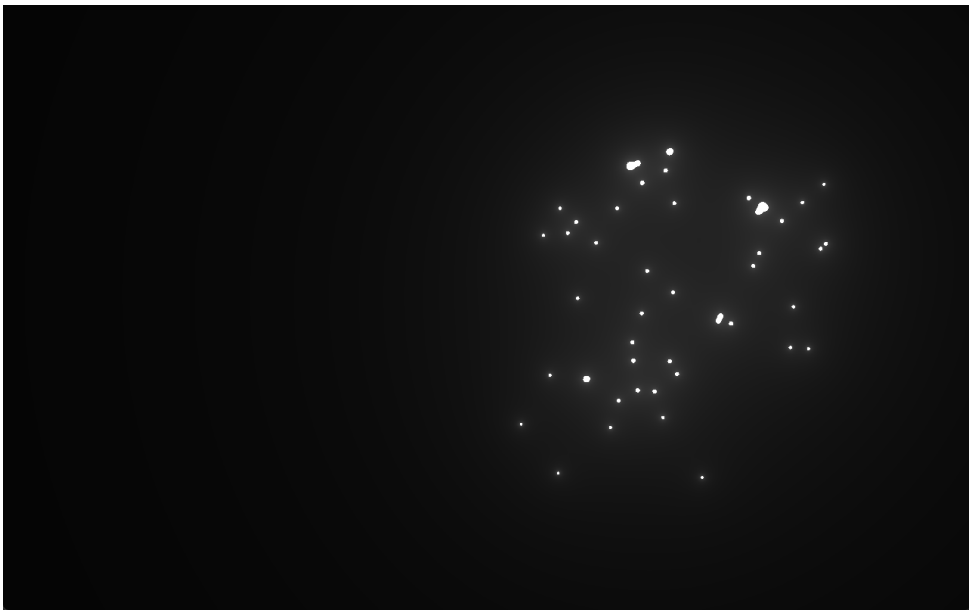
Obrázek 4.19: Příklad vykreslené animace v jazyce GLSL.

4.2.1 Světlušky začátek

Ve vystoupení se tato animace objeví úplně jako první. Jedná se tedy o něco jako seznámení s projekcí ale také s tanečníky. Ti se během této úvodní části shlukují na jednom místě jeviště. V animaci probíhá to samé. Padesát malých teček se začíná postupně objevovat na jedné straně projekční plochy. Postupně se zvětšuje jejich poloměr a začínají se prolínat mezi sebou.
Implementace:
Při tvorbě této animace bylo využito efektu zvaného **Metaballs** [26], který umožňuje organické spojování povrchu blízkých bodů v jeden celek.
Všech padesát bodů se náhodně pohybuje po dané ploše a zvětšuje se jejich poloměr až do určité velikosti. Při pohybu dochází k prolínání a spojování jejich povrchů. Projekce je promítána za tanečníky.



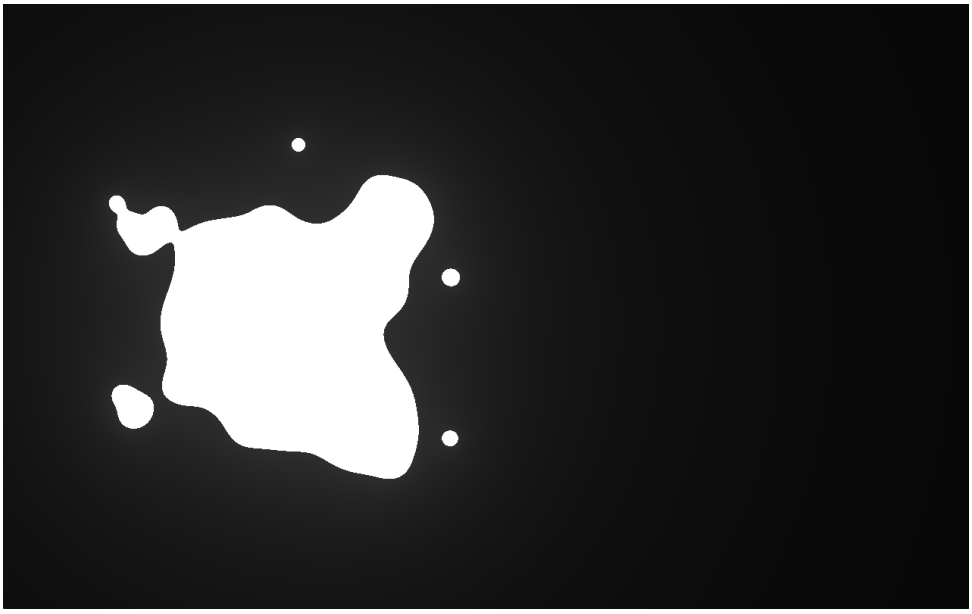
Obrázek 4.20: Tvorba efektu Metaballs. (převzato z [26].)



Obrázek 4.21: Vykreslená animace **Světlušky začátek**.

4.2.2 Světlušky konec

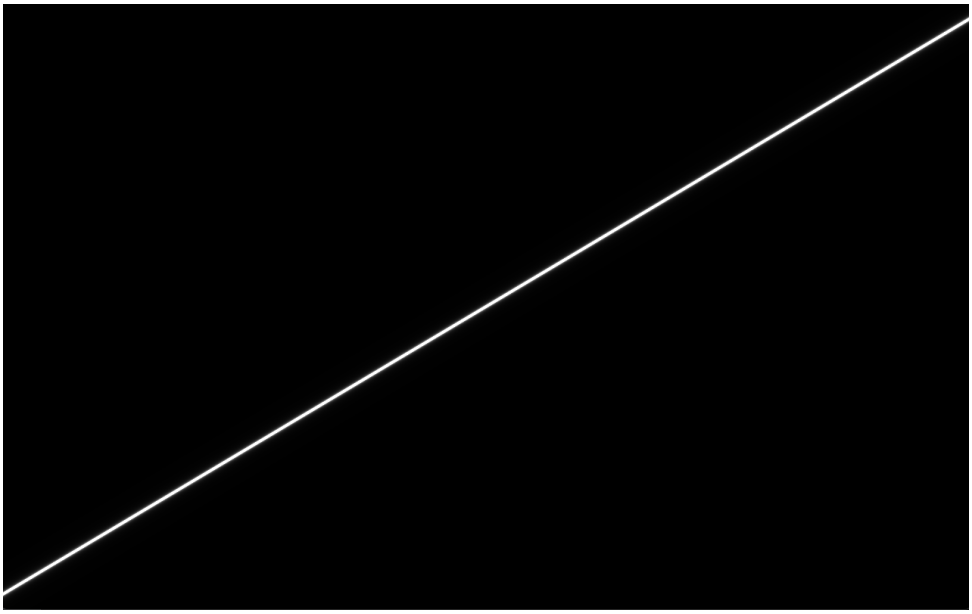
Tato animace je dokončením předchozí. Projekce je nyní promítána na zem vedle tanečníků. V průběhu animace se vykreslovaný shluk bodů přesouvá v souladu s hudbou z levé části projekční plochy na pravou. Vykreslované body se pak zvětšují a celá animace mizí do ztracena.



Obrázek 4.22: Vykreslená animace **Světlušky konec**.

4.2.3 Přímka

Animace přímky slouží jako sólo jednoho z tanečníků. Jedná se o představení tvaru přímky, která je jediným prvkem animace.
V jejím průběhu se přímka objeví přes celou projekční plochu. Následně se otáčí vlevo a vpravo podle hudby. Ke konci animace se přímka roztočí a mizí.
Implementace:
Přímka byla vytvořena podle návrhu v teoretické části (obrázek 2.6). V průběhu animace se mění úhel natočení obou bodů tak, aby se přímka otáčela na jednu nebo druhou stranu.

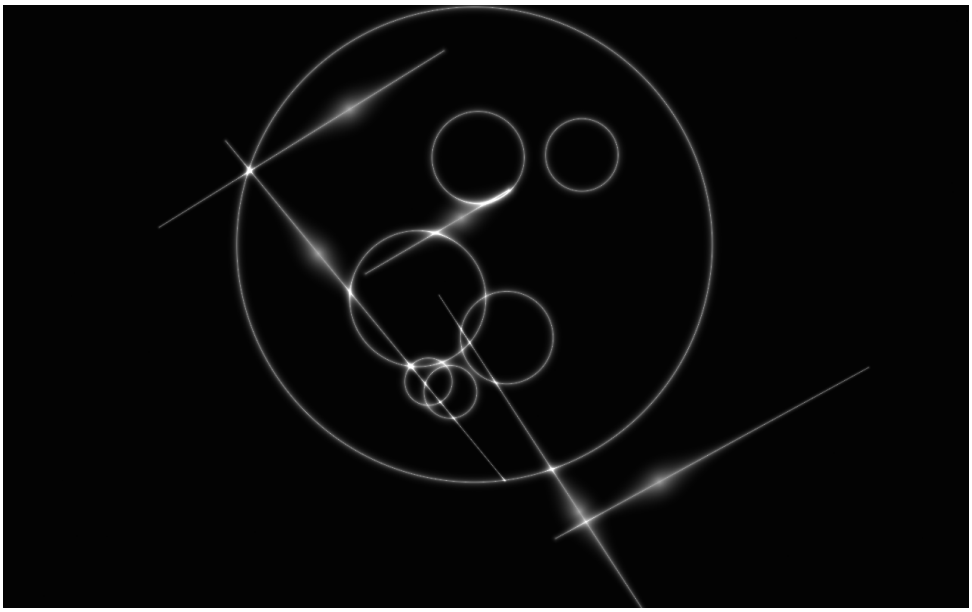


Obrázek 4.23: Vykreslená animace **Přímka**.

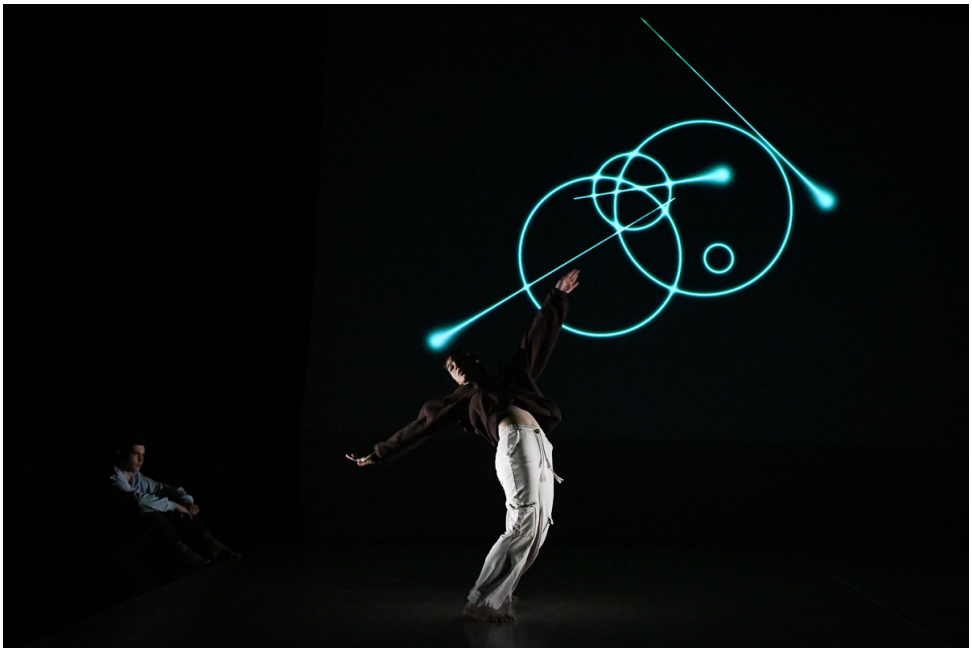
Obrázek 4.24: Animace **Přímka** na vystoupení.

4.2.4 Kruh a linka

Pomocí této animace byli představeni další dva tanečníci. Jedná se o prolnutí dvou stylů tance a také dvou tvarů v animaci. Ze začátku projekce se objevují úsečky a kružnice za tanečníkem a pohybují se po projekční ploše. Po určeném čase se projekce zastaví a přichází druhý tanečník. Projekce se opět začne pohybovat a ke konci mizí v levé straně projekce spolu s tanečníkem. Animace také spojuje předchozí animaci přímky a následující animaci kružnice.
Implementace:
Pro tuto animaci byly vytvořeny funkce vykreslující úsečku podobně jako v animaci přímky a kružnice pomocí středu a poloměru. Pro oživení byl na úsečku přidán pohybující se bod, který putuje po úsečce mezi jejími krajními body.



Obrázek 4.25: Vykreslená animace **Kruh a linka**.

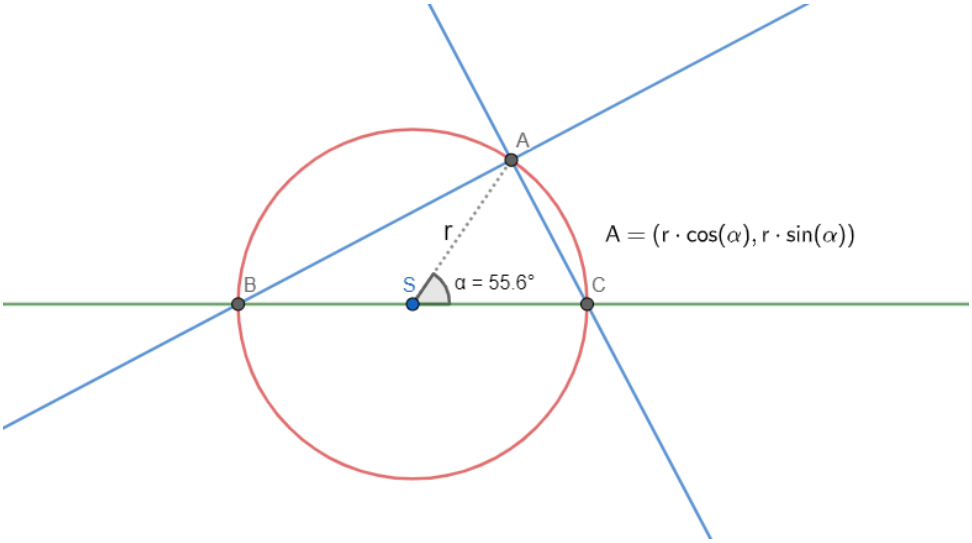


Obrázek 4.26: Animace **Kruh a linka** na vystoupení.

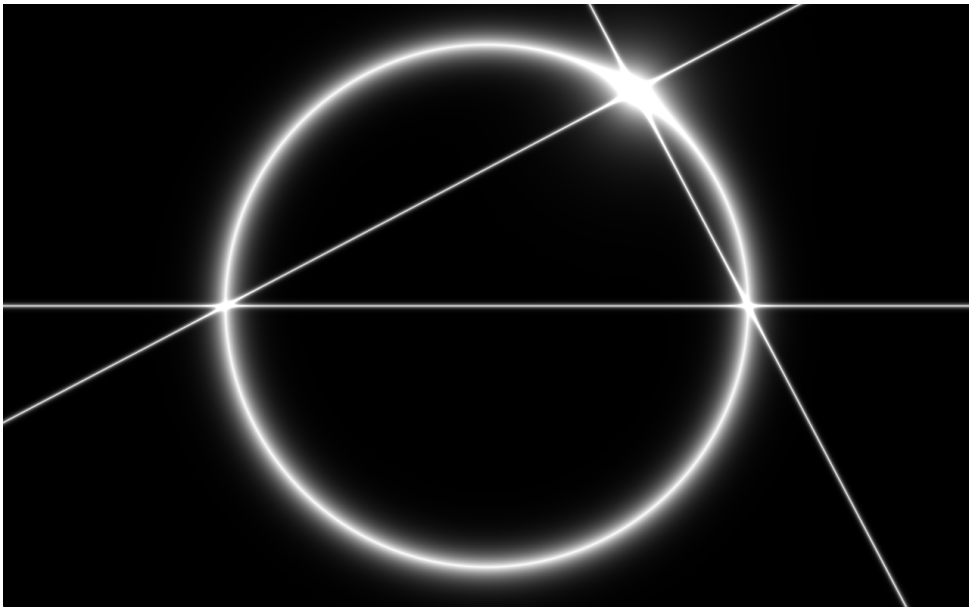
Obrázek 4.27: Animace **Kruh a linka** na vystoupení.

4.2.5 Kružnice

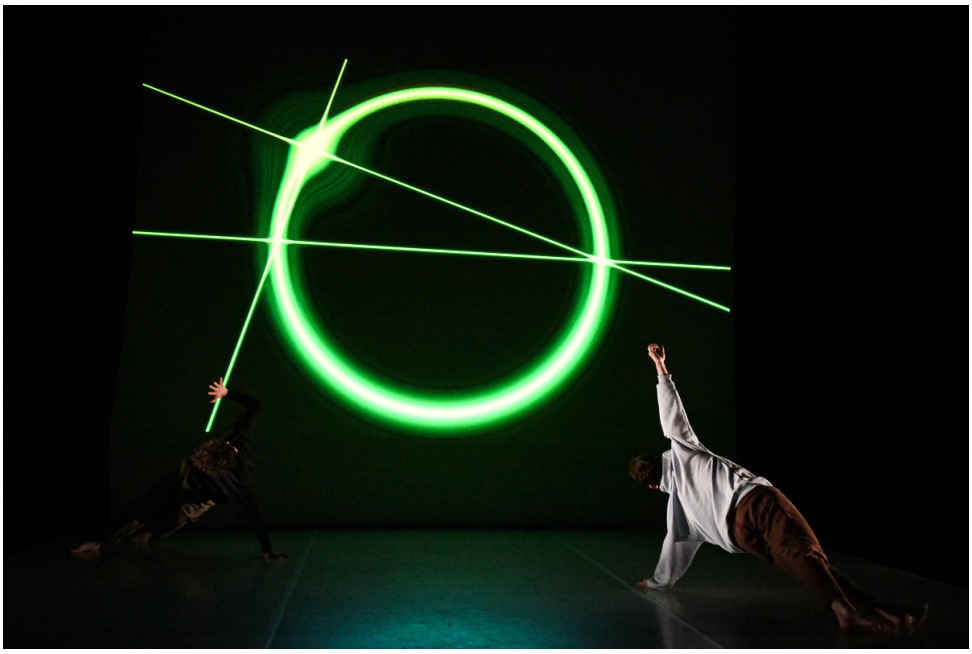
Po přechodu z předchozí animace se na projekční ploše objevuje velká kružnice vykreslená přes celou projekční plochu. Na kružnici je vykreslený jeden bod, který jí obíhá. Kružnice je také rozdělena přímkou procházející vodorovně přes její střed. Obíhající bod je spojen přímkami s průsečíky kružnice a dělící přímky. Vzniká tak trojúhelník vepsaný do oné kružnice, který se mění podle obíhajícího bodu.
Bod obíhá proti a následně po směru hodinových ručiček do hudby. Na konci animace se poloměr kružnice zmenšuje až do ztracena.
Implementace:
Kružnice je vytvořena opět podle návrhu z obrázku 2.5. Jsou zde ale navíc vykresleny přímky znázorněné na následujícím obrázku.



Obrázek 4.28: Návrh animace **Kružnice**.



Obrázek 4.29: Vykreslená animace **Kružnice**.

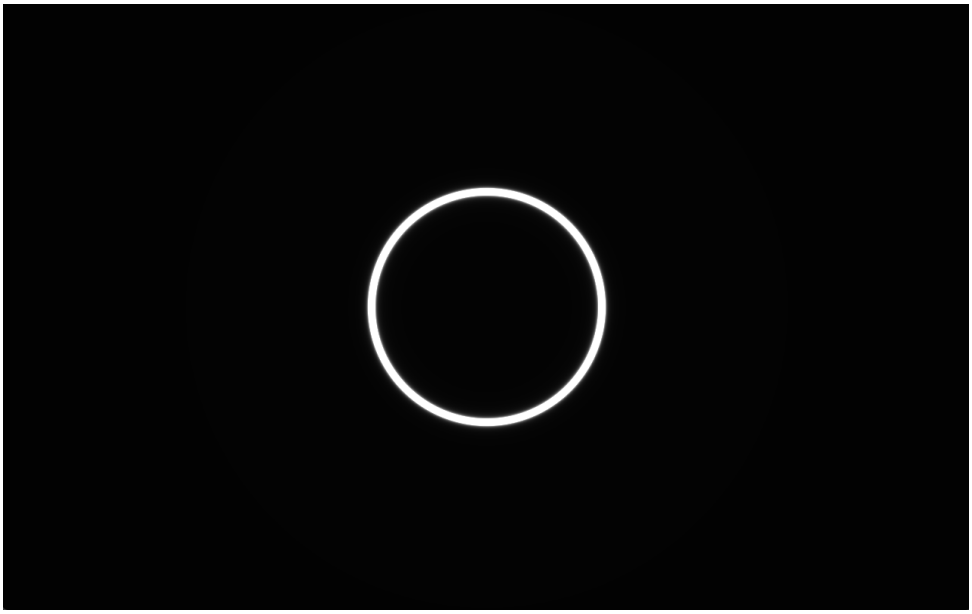


Obrázek 4.30: Animace **Kružnice** na vystoupení.

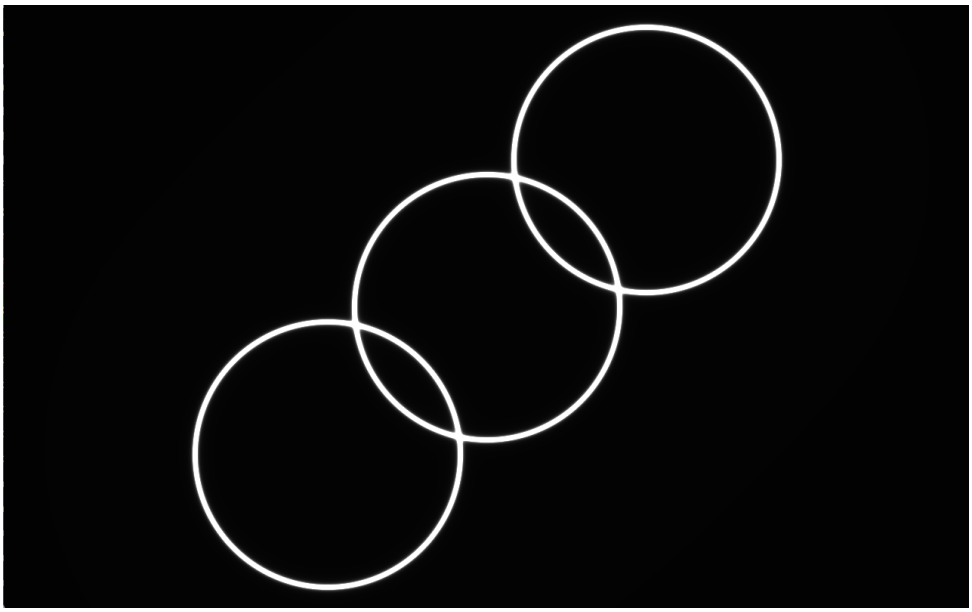
Obrázek 4.31: Animace **Kružnice** na vystoupení.

4.2.6 Kolečko

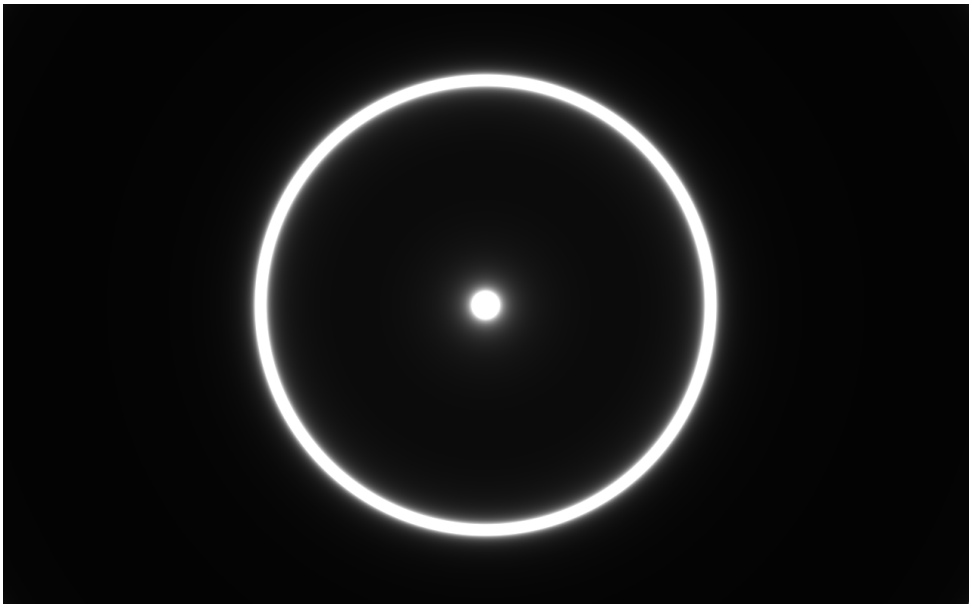
Druhou část celého představení uvádí tato animace, která je spojena ze tří podanimací prolínajících se mezi sebou. V první části se za tanečníky objevuje drobné kolečko, které se časem zvětší do tří čtvrtin velikosti celé projekce. Během zvětšování kolečka nastane okamžik, kdy se růst kolečka zastaví a celé kolečko se rozdělí na vícero stejných koleček, která se vzdálí od středu náhodně do strany a zpět. Reaguje na to jak hudba, tak i tanečníci. Po rozdělení se kolečka opět spojí. Po zvětšení kolečka nastává přechod v pulzy, které vycházejí od středu projekce až po její okraj. Pulzy jsou tvořeny opětovně se zvětšujícím se kolečkem. Na konci celé animace se pulzující kolečko zastaví a začíná se deformovat jako reakce na pohyby jednoho z tanečníků.
Implementace:
První část se zvětšujícím se kolečkem je opět vytvořena stejně jako v předchozích animacích, přičemž je ale umístěno více koleček na sebe tak, aby v daném okamžiku mohlo každé z nich provést posun mimo střed a zase se vrátit zpět na původní místo.
Při vzniku pulzů se původní kolečka vymažou a zachová se pouze jedno pulzující, které se vykresluje znovu a znovu vždy od nulového poloměru, který následně rychle roste. Rychlost pulzů byla stanovena tak, aby odpovídala hudbě.
Konečná deformace kolečka je způsobena deformací celého projekčního prostoru použitím Perlinova šumu [14], který náhodně ohýbá celou projekci.



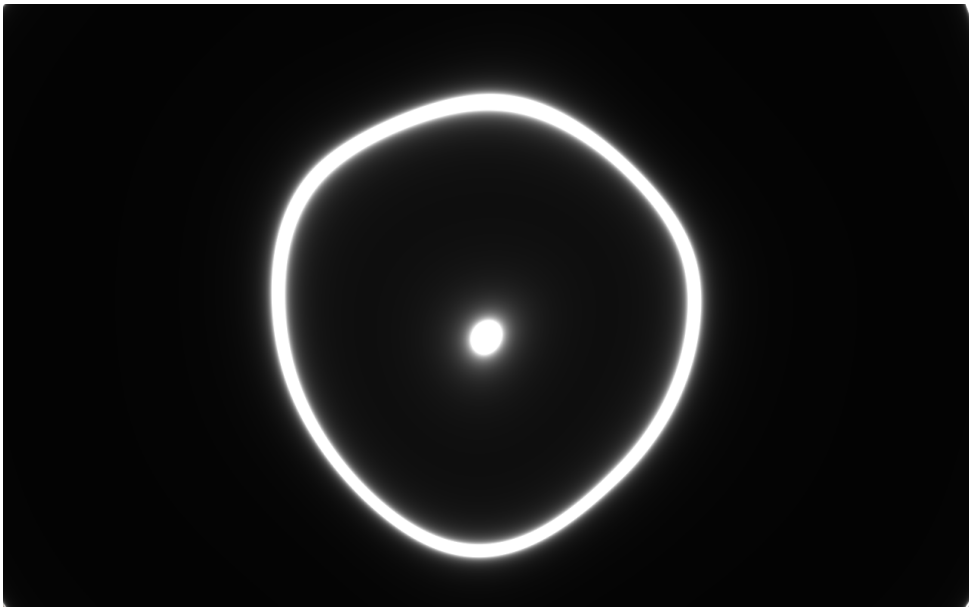
Obrázek 4.32: Vykreslená animace **Kolečko** (první část se zvětšujícím se kolečkem).



Obrázek 4.33: Vykreslená animace **Kolečko** (druhá část s rozpojením koleček).



Obrázek 4.34: Vykreslená animace **Kolečko** (třetí část s pulzy).



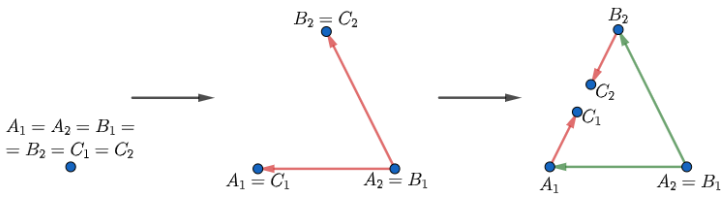
Obrázek 4.35: Vykreslená animace **Kolečko** (čtvrtá část s deformací kolečka).

Obrázek 4.36: Animace **Kolečko** na vystoupení.

Obrázek 4.37: Animace **Kolečko** na vystoupení.

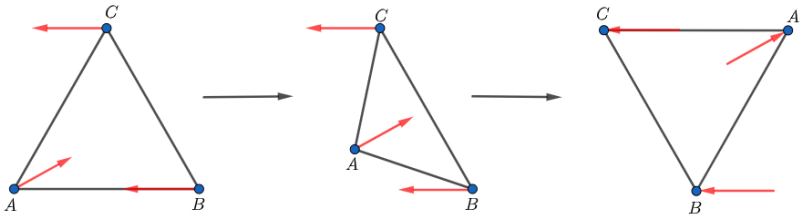
4.2.7 Trojúhelník

Třetí část vystoupení uvádí animace trojúhelníku, která vzniká na zemi pod jedním z tanečníků. Zbylí dva tanečníci přibíhají a ve stejnou chvíli se z bodu pod prvním tanečníkem začínají vzdalovat dvě úsečky, které se nakonec zastaví a třetí stranou vytvoří trojúhelník.
Vzniklý trojúhelník se pak dvakrát otočí v daný okamžik proti směru hodinových ručiček.
Po otočení se strany trojúhelníku třikrát prolnou. Dvakrát vždy jeden z vrcholů přejde na druhou stranu trojúhelníku, potřetí se prohodí všechny tři strany. Na konci animace se trojúhelník roztočí a zmenšuje se až do ztracena.
Implementace:
Celý trojúhelník má tři strany, které jsou definovány dvěma body. Celkem tedy máme šest bodů. Na počátku se body nacházejí na stejném místě v pravé části projekce tam, kde se nachází pravý vrchol trojúhelníku. Poté se vždy po dvojicích oddělují směrem ke zbývajícím vrcholům. Nakonec se poslední dva body dostávají na své místo a dotváří tak celý trojúhelník. Celý postup je dobře vidět na následujícím návrhu.



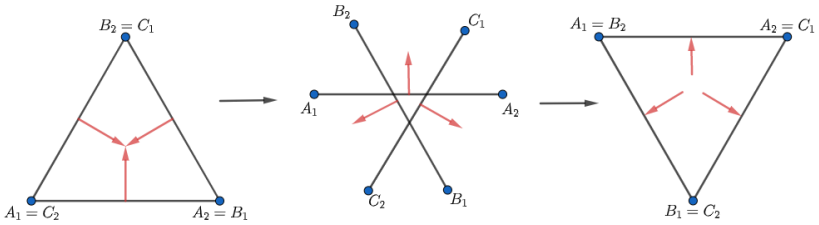
Obrázek 4.38: Návrh animace **Trojúhelník** (prvotní vykreslení).

Po dokončení této části se u již objeveného trojúhelníku prolnou dvakrát jeho vrcholy tak, jak je vidět na následujícím obrázku.



Obrázek 4.39: Návrh animace **Trojúhelník** (první dvě prolnutí).

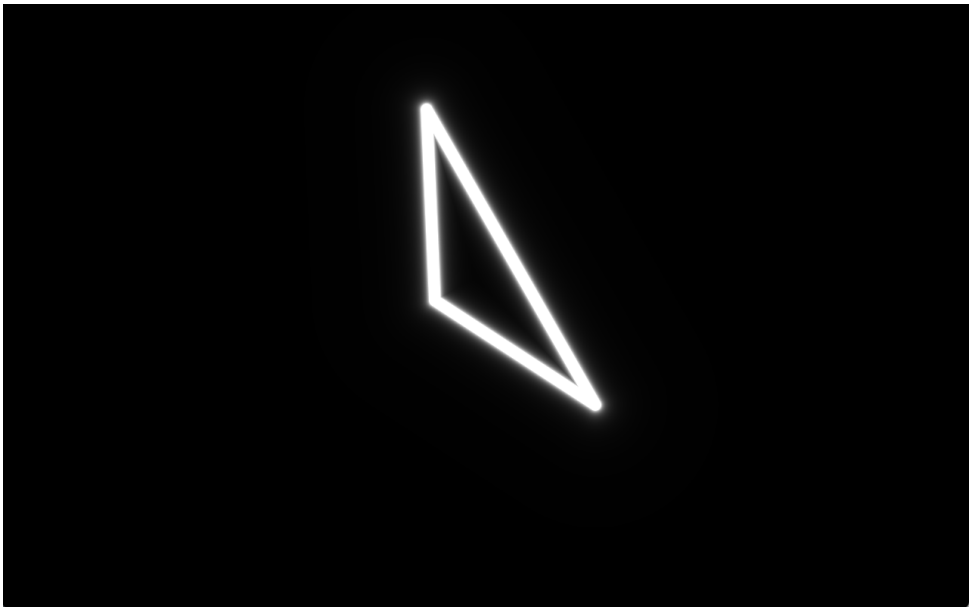
Třetí prolnutí probíhá následovně:



Obrázek 4.40: Návrh animace **Trojúhelník** (poslední prolnutí).

Po všech prolnutích se trojúhelník vykresluje již jen pomocí tří bodů tak, jak je definováno v teoretické části (obrázek 2.9).

Obrázek 4.41: Vykreslená animace **Trojúhelník** (prvotní vykreslení).



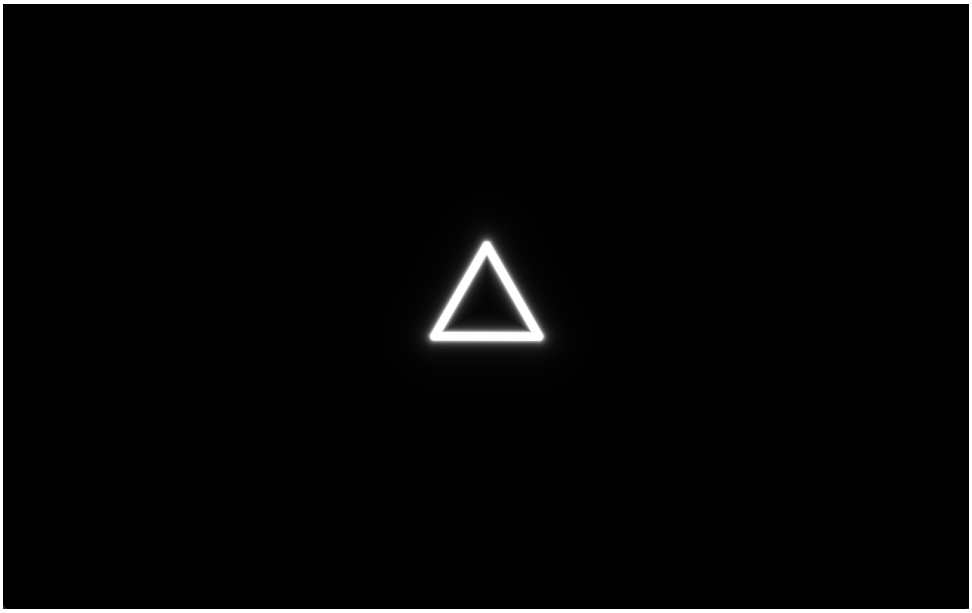
Obrázek 4.42: Vykreslená animace **Trojúhelník** (první dvě prolnutí).

Obrázek 4.43: Vykreslená animace **Trojúhelník** (poslední prolnutí).

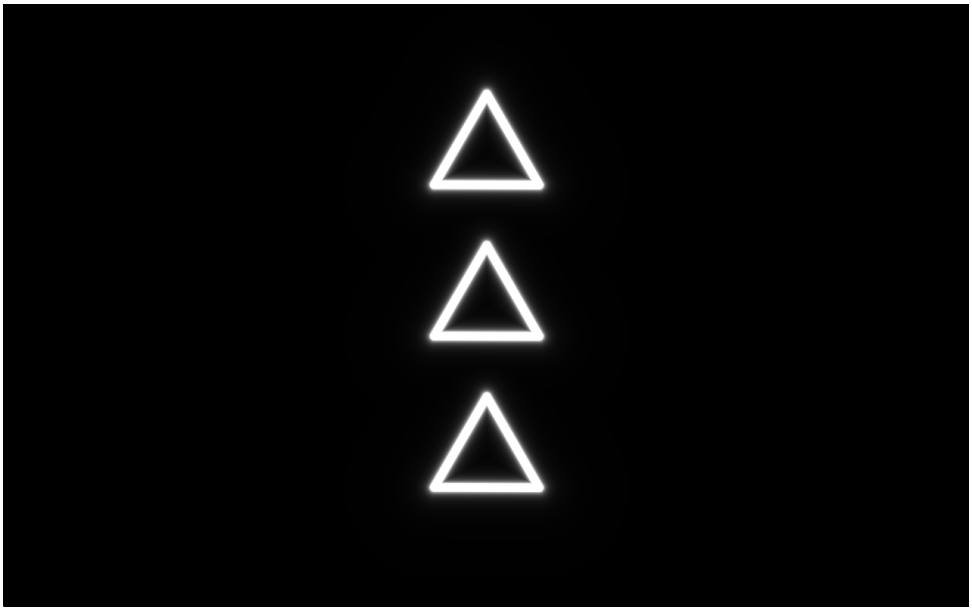
Obrázek 4.44: Animace **Trojúhelník** na vystoupení.

4.2.8 Tři trojúhelníky

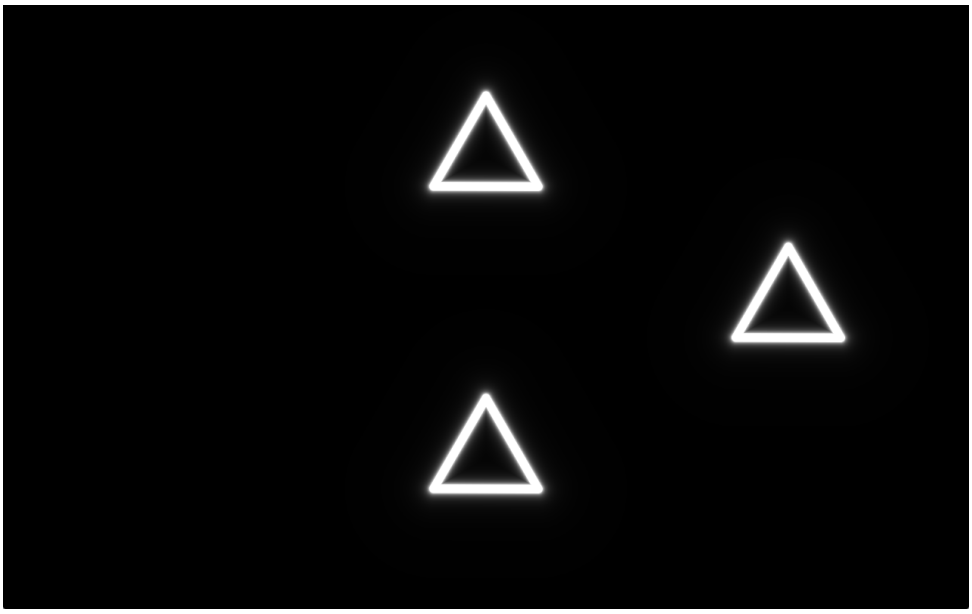
Tato animace kopíruje pohyby tří tanečníků. Nejprve tanečníci tvoří skupinu uprostřed jeviště. Za nimi se vykresluje trojúhelník. Tanečníci se po nějaké době seřadí za sebe a i trojúhelník se rozdělí na tři stejně velké nad sebou. V průběhu animace se prostřední trojúhelník posune vpravo stejně jako jeden z tanečníků. Na konci animace se trojúhelníky opět sjednotí v jeden.
Implementace:
Všechny tři trojúhelníky jsou tvořeny třemi vrcholy spojenými úsečkami podle návrhu z teoretické části (obrázek 2.9). Trojúhelníky jsou nejprve vykresleny přes sebe uprostřed projekce a až v zadaný čas se rozdělí. Řeší se tedy jen poloha jejich středu v čase.



Obrázek 4.45: Vykreslená animace **Tři trojúhelníky** (objevení).



Obrázek 4.46: Vykreslená animace **Tři trojúhelníky** (rozdělení).



Obrázek 4.47: Vykreslená animace **Tři trojúhelníky** (posun do strany).

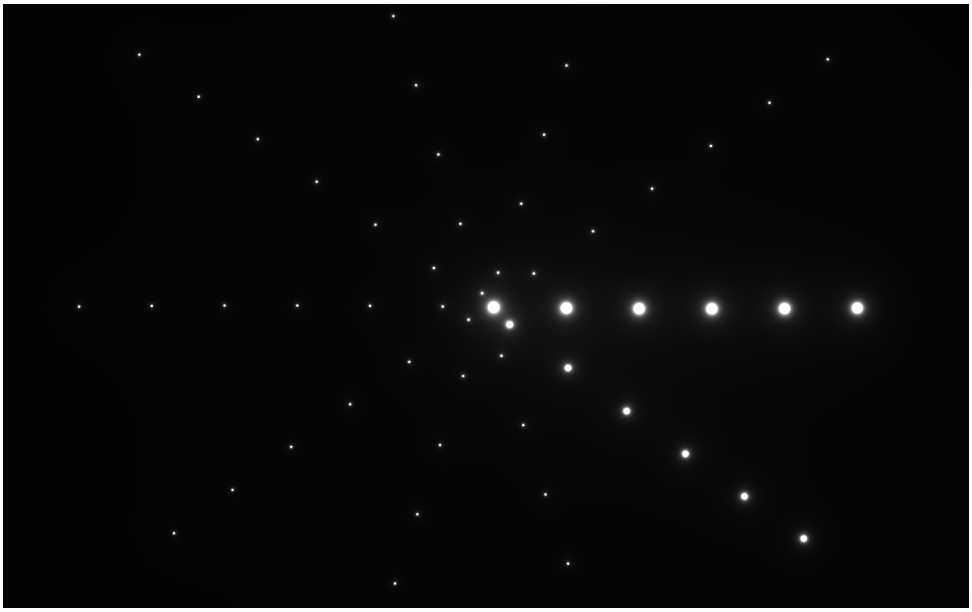
Obrázek 4.48: Animace **Tři trojúhelníky** na vystoupení.

4.2.9 Párty

Konečnou animací celého představení je tato animace. Má dotvářet atmosféru párty, která na jevišti vzniká. Animace znázorňuje řadu bodů uspořádaných od středu směrem vpravo za sebe. Body se točí rychlostí danou jejich vzdáleností od středu. Čím dále jsou, tím pomaleji se točí.
V průběhu animace se mění rychlost otáčení vzhledem k rychlosti tance tanečníků.
Implementace:
Každý z bodů má svůj poloměr, po kterém obíhá střed. Podle velikosti jeho poloměru se vypočítá jeho rychlost neboli úhel, o který se vždy pootočí. Body nejblíže středu obíhají nejrychleji.
Pokud se bod nachází v poloze jako na začátku, tedy vpravo od středu, je jeho intenzita zvýšena. Dochází tak k problikávání bodů v rámci jejich pohybu.
V průběhu animace také dochází ke zvětšení počtu bodů. To má za cíl vytvořit jakousi spirálu.

Obrázek 4.49: Vykreslená animace **Párty** (začátek animace).

Obrázek 4.50: Vykreslená animace **Párty** (tvorba spirály).



Obrázek 4.51: Vykreslená animace **Párty** (konec animace).

Obrázek 4.52: Animace **Párty** na vystoupení.

Kapitola 5

Testování

Kapitolu tvoří souhrn všech testů prováděných při tvorbě celého programu. Obsahuje testování celého programu z hlediska rychlosti a stability, testování detekčních algoritmů a také stability jednotlivých animací při vykreslování. Testy byly prováděny na dvou zařízeních a to notebooku MSI a stolním počítači. Jejich technické parametry jsou:

notebook:
Výrobce: *MSI*
Procesor: *Intel Core i5 2.9GHz*
Grafická karta: *Nvidia 940M*
Paměť RAM: *16GB*
Typ úložiště: *HDD*
Operační systém: *Linux Ubuntu 20.04*

stolní počítač:
Výrobce: *Custom*
Procesor: *Intel Core i7 3GHz*
Grafická karta: *Nvidia Geforce RTX 2060*
Paměť RAM: *16GB*
Typ úložiště: *SSD*
Operační systém: *Windows 11 Pro*

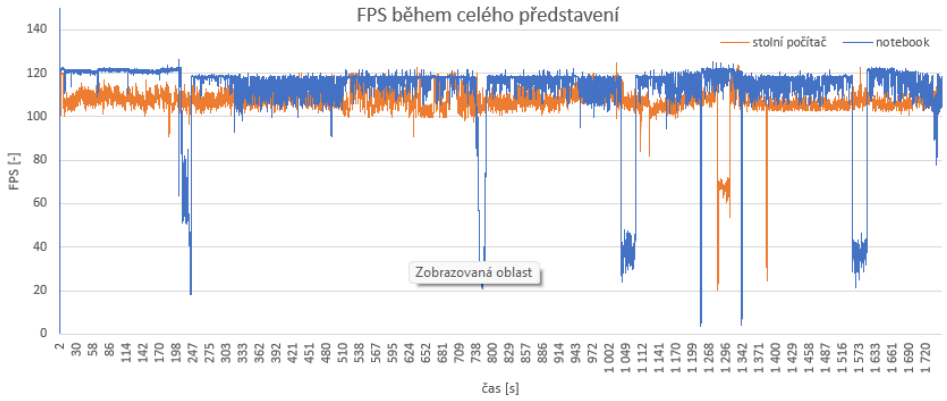
Testy byly prováděny i na notebooku, protože bylo potřeba zjistit, které části vystoupení jsou jak náročné. Jak je vidět z následujících grafů, poklesy u

snímkovací frekvence se výrazně projevily právě u notebooku a ne na stolním počítači. Cílem tedy bylo zjistit, co by se muselo zrychlit, aby se mohlo celé vystoupení pouštět právě na testovaném notebooku.

5.1 FPS

Počet snímků za sekundu je důležitým prvkem při testování. Je zde na první pohled jasné, jestliže je animace plynulá. Výrazně to pak ovlivňuje kvalitu celého vystoupení.

Pro testování FPS byla využita funkce knihovny Arcade, která umožňuje získat snímkovací frekvenci u právě vykreslovaného okna. Takto získané hodnoty byly ukládány v průběhu celého vystoupení do souboru. Výsledky jsou vidět v následujících dvou grafech.

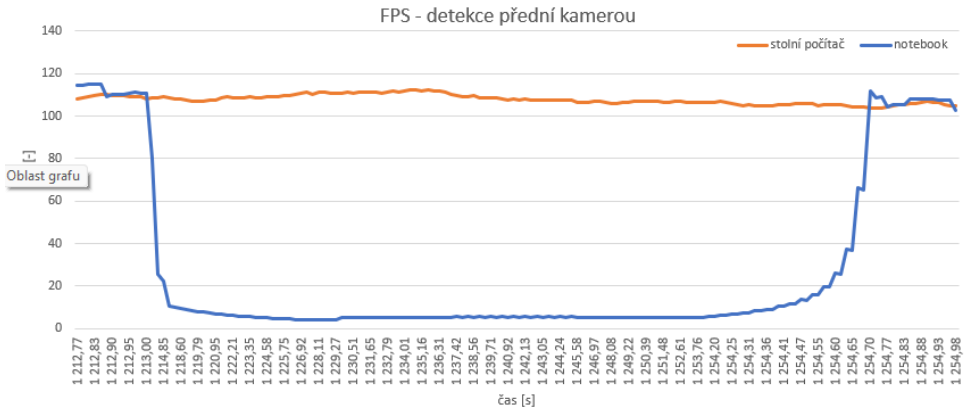


Obrázek 5.1: FPS v průběhu celého vystoupení.

Jak je z grafu vidět, většinu času se FPS drží nad 100 snímků za sekundu. U notebooku se ale v pár případech stal výrazný pokles. Ten je způsoben detekčními algoritmy nebo náročnější animací.
Z grafu je také vidět, že průměrná hodnota FPS u notebooku je 111 a u stolního počítače 107. Nižší hodnota FPS u stolního počítače je způsobena použitím operačního systému Windows.

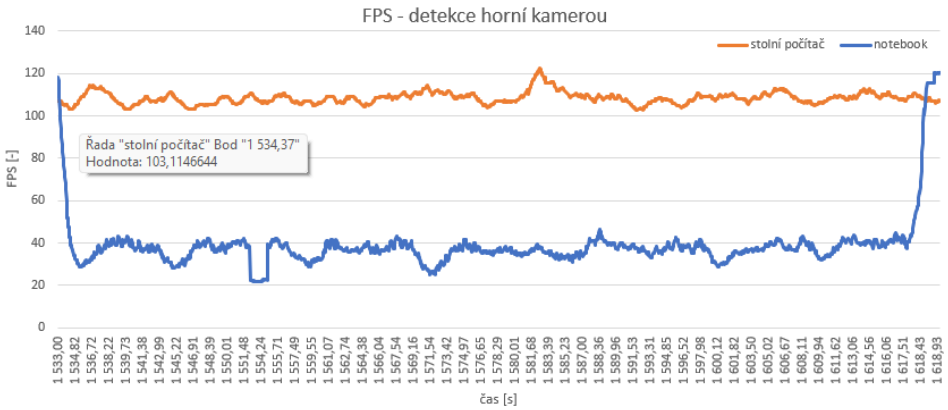
5.2 Rychlost detekce

Při spuštění detekčního algoritmu je nutné, aby během projekce nebyl počet snímků za sekundu nižší, než určitá mez, za kterou by již animace není plynulá.



Obrázek 5.2: FPS při detekci přední kamerou.

Z grafu je vidět, že FPS při detekci na notebooku klesly až na 5. Na stolním počítači se ale stále drží nad 100.
Detekční funkce je, jak již bylo řečeno v teoretickém rozboru, neuronovou sítí, která je již předem naučená. Při její implementaci je možné měnit jen pár parametrů jako je například vzdálenost detekované osoby od kamery. Nelze tu však změnit nic, co by výrazně snížilo výpočetní nároky.



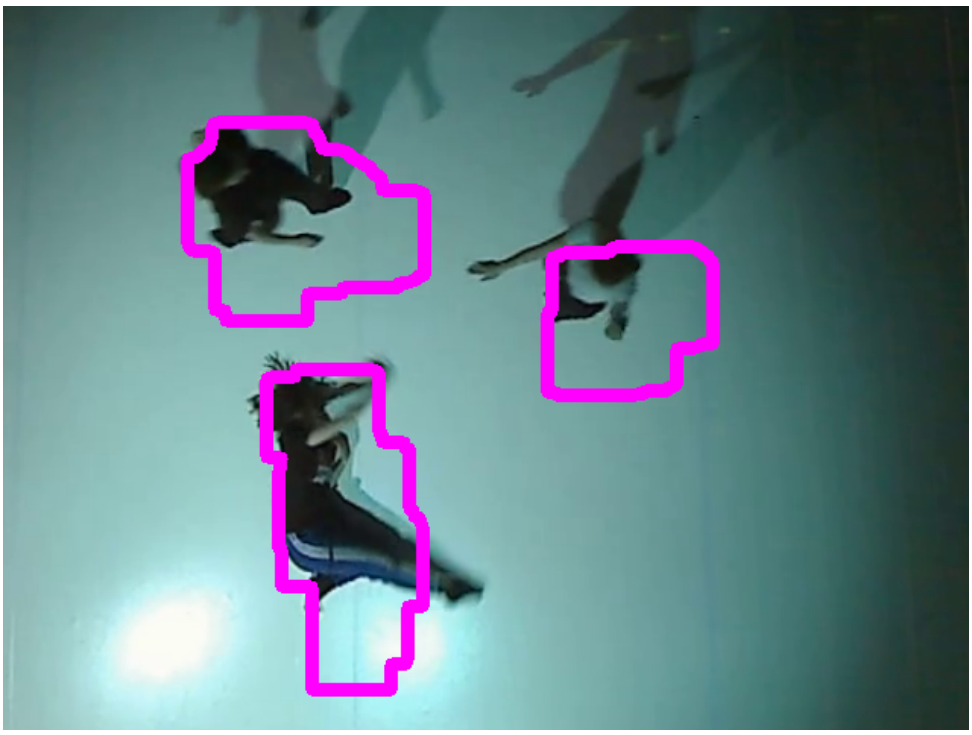
Obrázek 5.3: FPS při detekci horní kamerou.

U detekce přední kamery snímkovací frekvence u notebooku klesna nejméně na 20 snímků za sekundu.
Z obou grafů je vidět, že algoritmy počítající polohu tanečníků výrazně snižují snímkovací frekvenci. Je tedy velice důležité vybrat počítač se správnými komponenty, aby se celková FPS nedostala moc nízko.

5.3 Přesnost detekce

5.3.1 Detekce horní kamerou

Při detekci z horní kamery je obraz zpracováván pomocí funkcí knihovny OpenCV [21], která umožňuje detekovat kontury. Při implementaci byla otestována řada snímků, které zachycují detekované kontury. Středy detekovaných kontur byly porovnány s požadovanou hodnotou. Příklady detekce kontur jsou vidět níže:



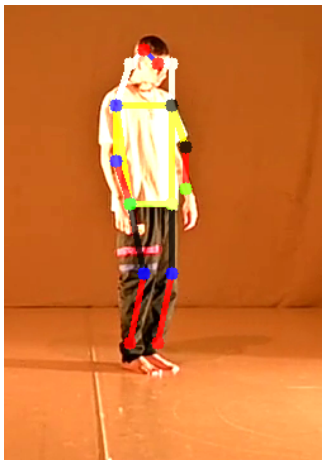
Obrázek 5.4: Příklad detekce kontur.

Obrázek 5.5: Příklad detekce kontur. (chybná detekce)

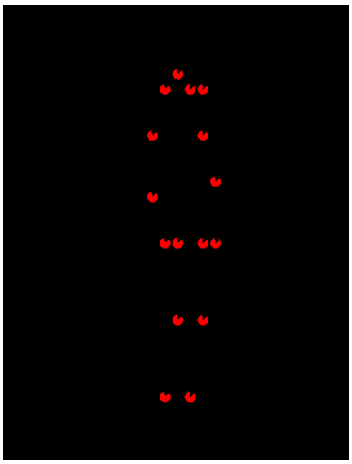
Z obou obrázků je vidět, že detekované kontury závisí jak na poloze tanečníka, tak i na správném osvětlení.
Pokud se tanečník nachází přímo pod kamerou, je detekce nejpřesnější. Čím dál do kraje se tanečník posune, tím více se projevuje perspektiva a detekovaná kontura je větší. Výsledná detekce je tedy méně přesná.
Při nesprávném osvětlení je také možnost detekovat kontury nesprávně jako je vidět na obrázku 5.5. Zde je chyba detekce způsobena špatným osvětlením místnosti.
Z opakovaného testování vyplývá, že pokud jsou tanečníci správně osvětleni a nestojí úplně na kraji detekované plochy, chyba detekce je kolem *20%*. Pokud se zhorší podmínky detekce, chyba roste až k *50%*.
Přesnost detekce je vyjádřena poměrem správných a chybných hodnot detekované polohy tanečníka.
V průběhu představení se bohužel nevyhneme chybě způsobené osvětlením, protože například u animace s názvem Pruhy, je do projekce zařazeno blikání celé projekční plochy. Toto blikání pak může způsobit chyby detekce.
Při tvorbě animací se tedy musí počítat s tím, že detekce nemusí být přesná a může se zde objevit i špatně detekovaná poloha. V animacích by tedy mělo být zahrnuto i toto riziko.

5.3.2 Detekce přední kamerou

Při detekci pomocí přední kamery byla testována jednak správná detekce (správně umístěné body na těle) a také odchylka umístěných bodů vzhledem k reálné poloze kloubu.

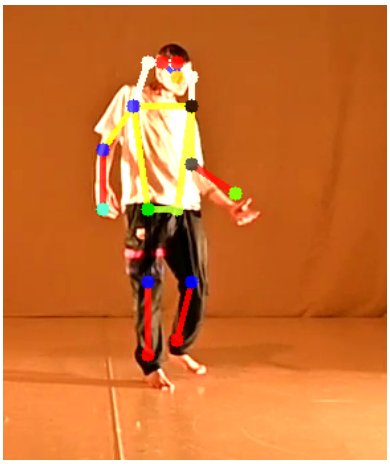


(a) : Obraz z kamery.

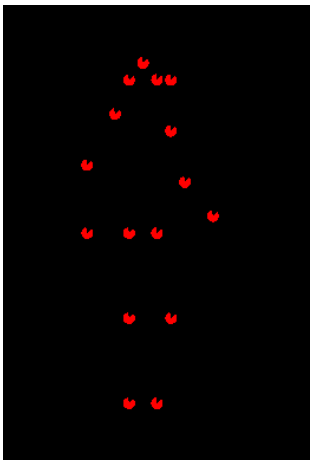


(b) : Samotné detekované body.

Obrázek 5.6: Příklad detekce pomocí přední kamery. (Snožmo)

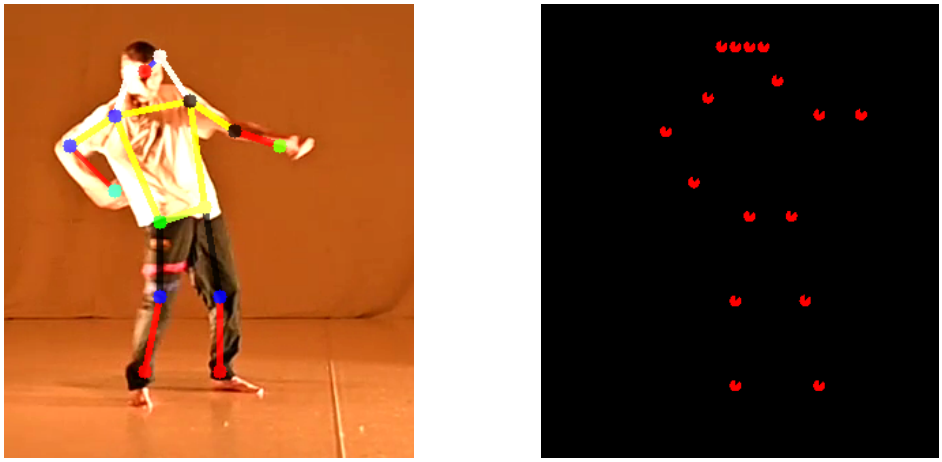


(a) : Obraz z kamery.



(b) : Samotné detekované body.

Obrázek 5.7: Příklad detekce pomocí přední kamery. (Odchylka v detekci zápěstí)



(a) : Obraz z kamery. **(b) :** Samotné detekované body.

Obrázek 5.8: Příklad detekce pomocí přední kamery. (Složitější poloha těla)

Z měření a testů vychází, že se detekované body od skutečných hodnot liší v řádu centimetrů.
Celková chyba detekce se pohybuje mezi *10% až 20%*.

5.3.3 Detekce gest

Při detekci gest se vychází z dat pořízených detekcí přední kamerou, tedy dat z neuronové sítě. Tato data mají chybovost kolem *15%*. Další odchylku pak přidá samotný algoritmus porovnávání pozic bodů lidského těla s databází gest. Celková chyba detekce gesta se tedy blíží k *30%*.
Při detekci gest ale nikdy nespouštíme vybranou animaci hned při prvním správně detekovaném gestu, ale vždy tanečník čeká určitou dobu, aby se gesto ověřilo. To minimalizuje chybu detekce a zabrání vykreslení animace při špatně detekované poloze tanečníka.

5.3.4 Podněty ke zlepšení

Po otestování funkcionality celého systému formou divadelního vystoupení byla zjištěna místa potencionálního zlepšení. Je to převážně detekce tanečníků pomocí horní kamery. Jak již bylo řečeno, tato detekce není prováděna pomocí neuronové sítě. Porovnáním s chybovostí pomocí přední kamery vyplývá, že přechodem na neuronovou síť by se chybovost jistě výrazně snížila.

Kapitola 6

Závěr

Prvním z cílů celé práce bylo seznámit se se stávajícími možnostmi grafické projekce při vystoupeních a současnými metodami detekce lidského těla. V teoretické části diplomové práce je této problematice věnována kapitola **Historie a současný stav**. Jsou zde shrnuty historické i současné metody používané při řešení této problematiky.
Hlavním z cílů diplomové práce bylo navrhnout systém umožňující projekci grafiky a snímání polohy tanečníků, pomocí níž se dá vykreslovaná grafika měnit v reálném čase. Návrh celého systému je popsán jak v **Teoretickém rozboru**, tak v kapitole **Praktické řešení**, kde je ukázána i jeho implementace.
Jednou z nejdůležitějších částí celé práce bylo implementovat jednotlivé animace vykreslované během vystoupení. Tyto animace jsou podrobně popsány v kapitole **Jednotlivé animace a jejich podrobný rozbor**.
Chod celého systému včetně všech jeho komponent byl opakovaně otestován jednak při samotné implementaci a také na patnácti zkouškách s tanečníky, které probíhaly od října loňského roku do letošního dubna. Výsledky testování jsou uvedeny v kapitole **Testování**.
Celý projekt vyvrcholil úspěšným představením **SHAPƎ** 2.5.2022 při představení bakalářských prací HAMU v divadle Ponec.

Příloha A

Literatura

[1] Martin A. Fischler a Robert A. Elschlager. „The Representation and Matching of Pictorial Structures“. In: *IEEE Transactions on Computers* C-22 (1973), s. 67–92.

[2] Yi Yang a Deva Ramanan. „Articulated pose estimation with flexible mixtures-of-parts“. In: *CVPR 2011* (2011), s. 1385–1392.

[3] Catalin Ionescu et al. „Human3.6M: Large Scale Datasets and Predictive Methods for 3D Human Sensing in Natural Environments“. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36 (2014), s. 1325–1339.

[4] Matthew Loper et al. „SMPL: a skinned multi-person linear model“. In: *ACM Trans. Graph.* 34 (2015), 248:1–248:16.

[7] Zhe Cao et al. „Realtime Multi-person 2D Pose Estimation Using Part Affinity Fields“. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2017), s. 1302–1310.

[9] Karim Iskakov et al. „Learnable Triangulation of Human Pose“. In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)* (2019), s. 7717–7726.

[10] Valentin Bazarevsky et al. „BlazePose: On-device Real-time Body Pose tracking“. In: *ArXiv* abs/2006.10204 (2020).

[11] Ardra Anilkumar et al. „Pose Estimated Yoga Monitoring System“. In: *SSRN Electronic Journal* (2021).

[12] Pierre Karashchuk et al. „Anipose: A toolkit for robust markerless 3D pose estimation“. In: *Cell reports* 36 (2021), s. 109730–109730.

Příloha B

Internetové odkazy

[5] Mourad Merzouki, Claire Bardainne a Adrien Mondot. *Performance of Pixel*. Youtube. 2015. URL: https://www.youtube.com/watch?v=pMuvrDwhp4w&t=104s&ab_channel=WIREDUK.

[6] SILA SVETA. *Amazing Dance and Video Mapping Performance*. Youtube. 2016. URL: https://www.youtube.com/watch?v=HB5nJB9R8Qw&t=33s&ab_channel=FunnyTv.

[8] Chris Cross. *Into the Light*. Youtube. 2018. URL: https://www.youtube.com/watch?v=ixgLqaidFwc&ab_channel=TEDxTalks.

[13] 3Blue1Brown. *Manim*. URL: https://github.com/3b1b/manim. (accessed: 01.05.2022).

[14] Khan Academy. *Perlin noise*. URL: https://www.khanacademy.org/computing/computer-programming/programming-natural-simulations/programming-noise/a/perlin-noise. (accessed: 01.05.2022).

[15] Ardra Anilkumar a Athulya K.T. a Sarath Sajan a Sreeja K.An. *Pose Estimated Yoga Monitoring System*. URL: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3882498. [cit. 30.01.2022].

[16] Arcade community. *Arcade*. URL: https://api.arcade.academy/en/latest/. (accessed: 01.05.2022).

[17] Pygame community. *Pygame*. URL: https://www.pygame.org/news. (accessed: 01.05.2022).

[18] Taichi community. *Taichi*. URL: https://github.com/taichi-dev/taichi. (accessed: 01.05.2022).

[19] *GLSL Shaders*. URL: https://developer.mozilla.org/en-US/docs/Games/Techniques/3D_on_the_web/GLSL_Shaders. (accessed: 01.05.2022).

[20] Christopher Jobson. *The Movement of Air*. URL: https://www.thisiscolossal.com/2015/11/movement-of-air-dance/. (accessed: 01.05.2022).

[21] OpenCV. *OpenCV*. URL: https://opencv.org/. (accessed: 01.04.2022).

[22] openVINO. *Human Pose Estimation Python* Demo*. URL: https://docs.openvino.ai/latest/omz_demos_human_pose_estimation_demo_python.html. (accessed: 01.04.2022).

[23] Inigo Quilez a Pol Jeremias. *Shadertoy*. URL: https://www.shadertoy.com/. [cit. 30.01.2022].

[24] Veronika Štefanová. *Laterna magika*. URL: https://vltava.rozhlas.cz/genius-posedly-divadlem-letos-si-pripominame-100-vyroci-narozeni-scenografa-8328618/2. (accessed: 01.05.2022).

[25] Olivia Ting. *Verses | SKETCH 9: Perspectives*. URL: https://olivetinge.com/Verses-SKETCH-9-Perspectives. (accessed: 01.05.2022).

[26] John Wigg. *Metaballs in 2D*. URL: https://john-wigg.dev/2DMetaballs/. (accessed: 01.05.2022).

Příloha C

Seznam zdrojových souborů

V odevzdané složce obsahující celý implementovaný program se vyskytuje řada souborů a podsložek. Zde je uveden jejich soupis a také popsaná jejich funkce.

- složka **human_pose_estimation_demo** a soubory **helpers.py, human-pose-estimation-0001.bin, human-pose-estimation-0001.xml, monitors.py** - soubory potřebné pro detekci částí těla neuronovou sítí
- složka **Prvky** - obsahuje všechny animace vytvořené pomocí knihovny *Arcade* a další pomocné komponenty například pro detekci gest tanečníků
- složka **shaders** - obsahuje všechny shadery napsané v jazyce GLSL
- složka **sprites** - obsahuje veškeré obrázky vykreslované jako sprity
- soubory **fps.txt, fps2.txt, FPS.xlsx** - obsahují data z měření FPS
- soubory **HcameraSetup.py, PcameraSetup.py** - komponenty starající se o oříznutí dat z kamer
- soubory **HcameraSetup.txt, PcameraSetup.txt** - data z předchozích komponent určující oříznutí dat z kamer
- soubor **Hudba.wav** - audiosoubor s hudbou pro vystoupení
- soubor **main.py** - hlavní soubor pro spuštění programu
- soubor **timeline.txt** - soubor pro časové řízení představení
- soubor **topdown.mp4** - videozáznam z horní kamery pro testovací účely

Příloha D

První spuštění

Před prvním spuštěním programu je potřeba nainstalovat si potřebné balíčky pro Python.
Seznam balíčků i s příslušnými verzemi je následující:

```
1 Pillow 9.0.1
2 arcade 1.8.0
3 numpy 1.19.5
4 opencv-python 4.5.5.64
5 openvino 2021.4.2
6 pyglet 2.0a2
7 scipy 1.8.0
```

Listing D.1: Soupis knihoven potřebných pro chod programu.

ZADÁNÍ DIPLOMOVÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Tyle** Jméno: **Ondřej** Osobní číslo: **437479**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra řídicí techniky**

Studijní program: **Kybernetika a robotika**

Studijní obor: **Kybernetika a robotika**

II. ÚDAJE K DIPLOMOVÉ PRÁCI

Název diplomové práce:

Grafický doprovod tanečního představení

Název diplomové práce anglicky:

Graphic accompaniment of a dance performance

Pokyny pro vypracování:

1) Prostudujte současné technologie a metody sledování pohybu lidské postavy v kamerovém záznamu a zhodnoťte je z hlediska rychlosti a přesnosti.
2) Na základě výsledků z předchozího semestrálního projektu specifikujte problém spojený se sledováním pozice tanečníka na ploše a s generováním grafických objektů promítaných na plochu podlahy a stěny v reakci na jeho pohyb. Navrhněte řetězec, který bude tuto úlohu provádět na základě předem daného scénáře definujícího reakce vizuálního výstupu na pohyb několika tanečníků současně.
3) Dále navrhněte prostředí pro definici a spouštění sekvencí, při nichž na základě výše zmíněného časového scénáře bude možné vybranou figurou každého tanečníka vždy spouštět některou kresbu grafického objektu.
4) Implementujte výše popsaný řetězec ve formě aplikace s uživatelským rozhraním. Ověřte funkčnost a přesnost kresby grafických objektů v reálných podmínkách.
5) Proces návrhu, testování, použití a kalibrace zdokumentujte.

Seznam doporučené literatury:

[1] Anilkumar, A., K.T., A., Sajan, S. and K.A., S., Pose Estimated Yoga Monitoring System. Proceedings of the International Conference on IoT Based Control Networks & Intelligent Systems - ICICNIS 2021, Available at SSRN: https://ssrn.com/abstract=3882498 or http://dx.doi.org/10.2139/ssrn.3882498.
[2] Bazarevsky, V., Grishchenko, I., Raveendran, K., Zhu, T., Zhang, F. and Grundmann, M., BlazePose: On-device Real-time Body Pose tracking. arXiv preprint arXiv:2006.10204, 2020.

Jméno a pracoviště vedoucí(ho) diplomové práce:

Ing. Roman Berka, Ph.D. Katedra počítačové grafiky a interakce

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) diplomové práce:

Datum zadání diplomové práce: **31.01.2022** Termín odevzdání diplomové práce: **20.05.2022**

Platnost zadání diplomové práce:
do konce letního semestru 2022/2023

Ing. Roman Berka, Ph.D.	prof. Ing. Michael Šebek, DrSc.	prof. Mgr. Petr Páta, Ph.D.
podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Diplomant bere na vědomí, že je povinen vypracovat diplomovou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací.
Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v diplomové práci.

. _____
Datum převzetí zadání

_____
Podpis studenta