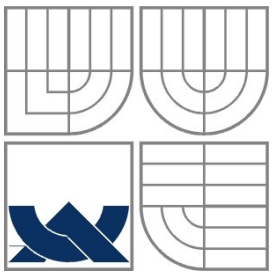
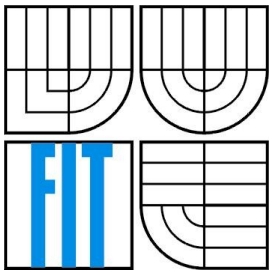


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

PROMÍTANÉ UŽIVATELSKÉ ROZHRANÍ – KARETNÍ HRA

PROJECTED USER INTERFACE – CARD GAME

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

TOMÁŠ BRESTIČ

VEDOUCÍ PRÁCE
SUPERVISOR

ING. PAVEL NAJMAN

BRNO 2016

Abstrakt

Ve snaze o zvýšení intuitivnosti používání počítače jsou zkoumány možnosti nahrazení nutnosti práce se vstupními zařízeními uplatněním gest k jeho ovládání a stále častěji se objevují pokusy o realizaci tohoto přístupu. Současně se vyskytuje tendence začleňovat výstup programu do fyzického světa kolem uživatele a celkově tak minimalizovat všechny umělé prostředky interakce. Cílem této práce je implementovat karetní hru s využitím zmíněných principů. Zabývá se zejména problematikou snímání pohybů člověka, detekce rukou a rozpoznávání jejich gest. Výsledná aplikace používá hloubková data z druhé generace senzoru Kinect, analytický popis gest a dovoluje hru až čtyř hráčů. Její grafický výstup je promítán na stůl za účelem zvýšení podobnosti s reálným prostředím.

Abstract

Current effort to make human-computer interaction more intuitive leads to utilization of gestures replacing the need to use a conventional input device. At the same time there is a tendency to integrate the output into the physical world around the user and therefore to minimize any artificial means of interaction in general. The aim of this thesis is to implement a card game that can be controlled on the basis of the principles described above. Fundamental issues are human motion tracking, hand detection and gesture recognition. The application uses depth data from the second generation of Kinect sensor, analytical description of gestures and allows up to four players to play the game. Its graphics are projected on the table to maximize compliance with the real situation.

Klíčová slova

uživatelské rozhraní, HCI, NUI, tabletop, detekce ruky, rozpoznávání gest

Keywords

user interface, HCI, NUI, tabletop, hand detection, gesture recognition

Citace

BRESTIČ, Tomáš. *Promítané uživatelské rozhraní – karetní hra*. Brno, 2016. 36 s. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Najman Pavel.

Promítané uživatelské rozhraní – karetní hra

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením Ing. Pavla Najmana. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Tomáš Brestič
v Brně, 16. 5. 2016

Poděkování

Děkuji Ing. Pavlu Najmanovi za trpělivé vedení a cenné odborné rady, které mi poskytl při tvorbě této práce.

Obsah

1 Úvod

Obecným cílem oblasti uživatelských rozhraní je dosažení co možná nejintuitivnějšího způsobu interakce člověka se strojem, počítačem, za účelem zvýšení efektivity jeho využití při současné maximalizací pohodlí uživatele. Tento dlouhodobý trend se v informatice projevuje jednak postupným zvyšováním abstrakce a zpřehledňováním ovládání, např. od příkazové řádky (angl. *command-line interface*, CLI) k tlačítkům, ikonám a oknům (grafické uživatelské rozhraní, angl. *graphical user interface*, GUI), a také rostoucí ergonomií vstupních zařízení, např. použití počítačové myši.

Dalším krokem v této snaze je použití tzv. přirozeného uživatelského rozhraní (angl. *natural user interface*, NUI), jenž spočívá v minimalizaci umělých vizuálních ovládacích prvků a ve využití snímání pohybů člověka k ovládání aplikace. Navržené ovládací pohyby by přitom měly být co nejpřirozenější, v optimálním případě mohou odpovídat gestům, která se mimoděk vyskytují v mezilidské komunikaci či při běžné manipulaci s reálnými, fyzickými objekty. Další takovou tendencí je snaha začlenit grafický výstup z počítače do reálného světa kolem uživatele. S tím souvisí pojem rozšířená realita (angl. *augmented reality*, AR). Tato práce si klade za úkol navrhnout a realizovat počítačovou karetní hru s rozhraním založeným na těchto principech, seznámit čtenáře se zvolenými postupy a zhodnotit jejich vhodnost a účelnost.

Základním tématem řešené problematiky je ovládání samotné, tedy detekce ruky a rozpoznávání gest. Obsahem práce je ale i vytvoření takto ovládané aplikace v podobě karetní hry, neboli naprogramování herní logiky a zpracování grafického výstupu. V následujících kapitolách budou postupně představeny některé stávající přístupy, technologie a programová prostředí, která lze k řešení uvedeného problému použít, dále bude navržena konkrétní podoba teoretického řešení za využití prostředků z nich vybraných a později bude následovat podrobnější popis vytvořené praktické implementace a její kvalitativní vyhodnocení.

2 Teorie

Účelem této kapitoly je seznámit čtenáře s vybranými existujícími prostředky vhodnými k tvorbě aplikace s ovládáním pomocí gest ruky. Práce si nedělá ambice být kompletním výčtem všech možných řešení veškerých myslitelných problémů, jde spíše o přehled několika typických přístupů k základním otázkám, které s jejím tématem souvisí, se zvláštním důrazem na ty z nich, které budou použity při realizaci aplikace. Je uplatněna snaha o stručný a pochopitelný nástin principu jednotlivých metod, přičemž bývá uvedena literatura pokrývající příslušné eventuální formalismy. Teoretický koncept je obvykle doplněn příkladem praktického výskytu či použití.

2.1 Tabletop

Dle zadání této práce má být grafický výstup aplikace vykreslen na stůl. Se zobrazenou informací pak mají uživatelé interagovat přímo – nikoliv prostřednictvím dalšího zařízení (klávesnice, myš), ale pomocí vlastních gest. Taková sestava patří do obecnější skupiny interaktivních zařízení využívajících horizontální plochy k zobrazování dat, která se souhrnně označuje jako tabletops (z angl.; termín přejímá svoji stavbu z dřívějších pojmů *desktop* a *laptop*). Tento přístup je jedním ze způsobů kombinace fyzického prostředí s virtuálním prostorem a je obzvláště výhodný pro využití menší skupinou několika uživatelů současně. [1]

Pro realizaci vlastního zobrazování i ovládání se používají různé přístupy. Lze použít např. přední (shora) či zadní (zespodu) projekci, LCD nebo OLED displeje. Vstupy od uživatele mohou být zprostředkovány skrze dotykové rozhraní či pomocí snímání jeho pohybů, případně může po stole pohybovat speciálními předměty, jež jsou rozpoznávány a sledovány, např. za použití markerů. Každou z uvedených variant lze navíc implementovat hned několika způsoby [1, 2]. Detailněji bude popsán princip snímání pohybů člověka, neboť právě ten je relevantní k dalšímu obsahu práce.

2.2 Senzory pohybu člověka

Má-li být aplikace ovládána pohyby člověka, je třeba tyto pohyby vhodným způsobem snímat. Zařízení k tomu určená mohou fungovat na mnoha rozličných fyzikálních principech, z nichž vyplývají výhody i omezení, která se s jejich použitím pojí. Liší se počet sledovatelných stupňů volnosti (až 6 – pozice a orientace ve třech osách), přesnost, snímací frekvence, prostorový rozsah oblasti pro sledování, náchylnost k vlivu šumu, nároky na konstrukci, cena a bezpočet dalších, specifičtějších parametrů. K optimalizaci těchto kritérií vzhledem k potřebám konkrétní aplikace je možné kombinovat i více přístupů najednou.

2.2.1 Mechanické snímání

Mechanické snímání využívá fyzické spojení sledované části těla systémem kloubů a táhel s pevnou konstrukcí vybavenou mechanoelektrickým převodníkem, typicky s pomocí nějakého druhu potenciometru. Tento přístup dovoluje sledovat pohyb cíle vůči dané konstrukci s vysokou přesností, pohyb samotný je však podstatně omezen možnostmi použitých ramen, jež jej převádějí [2]. Příkladem může být systém Fakespace Binocular Omni-Orientation Monitor (BOOM) nebo část herního ovladače konzole NES Mattel/PAX Power Glove. Konstrukci lze využít nejen k získání vstupů aplikace, ale též k obohacení výstupů směrem k uživateli – může klást pohybu proměnný odpor, vibrovat a podobně, a tím zprostředkovávat hmatový vjem virtuálního prostředí (angl. *haptic devices*).

2.2.2 Inerciální snímání

Prostorové inerciální snímání uplatňuje 3 gyroskopy ke zjištění natočení a 3 akcelerometry ke získání zrychlení v jednotlivých směrech. Z těchto informací lze po odečtení vlivu gravitace zjišťovat polohu zařízení, resp. části lidského těla, na níž je upevněno. V tomto případě není nutné sledovaný cíl spojovat s prostředím soustavou mechanických komponent přenášejících jeho pohyb, nemusí být zajištěn ani přímý výhled na něj, jako tomu bude u některých dalších typů senzorů. Přístup je též odolný vůči vlivům elektromagnetických vlastností prostředí. Nevýhodou je to, že i malá nepřesnost některé součástky způsobuje znatelný postupný posun vypočtené pozice od pozice skutečné [2]. Senzor navíc musí být umístěn přímo na části těla, jejíž pohyb chceme sledovat, což může být problém, je-li vyžadována zvlášť vysoká přesnost, neboť hmotnost takových snímačů je často značná.

2.2.3 Magnetické snímání

Magnetické snímání funguje na principu porovnání náklonů magnetických senzorů v jednotlivých osách k referenčnímu magnetickému poli. Tím může být magnetické pole Země nebo pole uměle vytvářené nějakým aktivním zdrojem. Při použití aktivního zdroje magnetického pole lze přepínat mezi několika cívkami, tím ovlivňovat tvar generovaného magnetického pole a snížit tak počet nutných magnetických čidel. Je tím však omezena maximální vzdálenost od zdroje, ve které lze pohyb detekovat, resp. přesnost se s rostoucí vzdáleností od zdroje rapidně snižuje [2]. Tento způsob je dále citlivý na magnetické vlastnosti okolí, jinak však dovoluje podobnou volnost pohybu jako použití inerciálních senzorů. Přístup je využit například u zařízení Flock of Birds firmy Ascesion.

2.2.4 Optické snímání

Pod pojmem optické snímání je zastoupeno množství různých principů k odvození informace o pohybu z detekce různých vlastností světla, zejména z jeho barvy nebo intenzity. Světlo přitom může být uvnitř i vně viditelného spektra, přístup může i nemusí přímo zahrnovat speciální zdroje světla (ovládané v rámci snímání), různé kombinace umístění těchto zdrojů vůči senzorům (*outside-looking-in*, *inside-looking-out*), může být využito analogových i digitálních zařízení, před něž mohou být umístěny filtry, apod. [2]

Nejjednodušším případem je použití fotodiody či fotorezistoru, tedy elektronických součástek ovlivňujících proud, který přes ně prochází, na základě množství dopadajícího světla. Za použití většího množství takových součástek lze například zjistit, kam dopadá nejvíce světla v rámci dané plochy. Na tomto principu fungují PSD (z angl. *position sensitive device*).

Dále lze použít např. technologii CCD (z angl. *charge-coupled device*), která převádí světelnou energii na elektrický náboj. Po jeho navzorkování mohou být výsledkem data reprezentující např. barevný obraz. Z několika takových vzájemně posunutých obrazů lze pomocí triangulace získat prostorovou informaci.

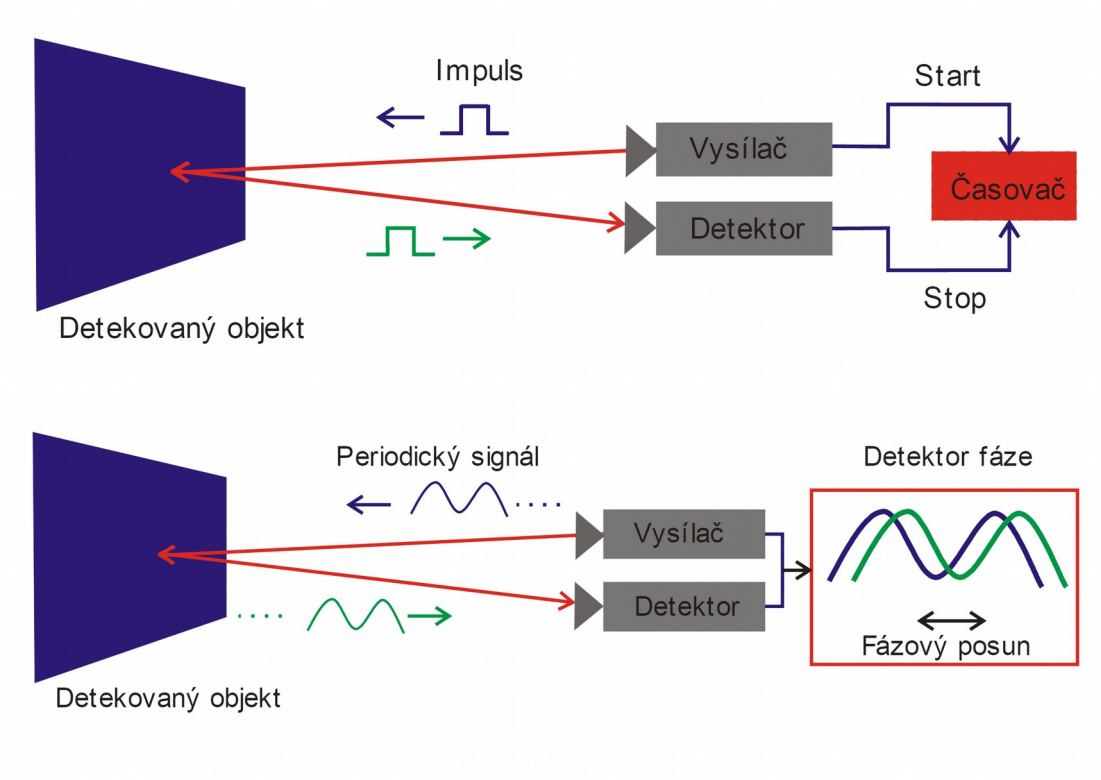
Je-li součástí senzoru vhodný zdroj světla, lze ke zjištění prostorových vlastností objektu použít techniku *Structured Light Imaging*, která zkoumá zkreslení předem známého vzoru promítnutého na nerovný předmět. Takto zjišťuje vzdálenost např. senzor Kinect první generace.

Společnou vlastností všech zmíněných metod je potřeba nějakého osvětlení a přímé viditelnosti ze senzoru na sledovaný cíl. Jejich výsledky jsou dále ovlivněny barevností sledovaných objektů a pozadí.

2.2.5 Elektromagnetické ToF snímání

Tento typ senzorů vysílá paprsky elektromagnetického záření z obecně širšího rozsahu než předchozí optické metody, tedy včetně radiových a mikrovlnných pásem. Zjišťuje čas, za který se paprsky odrazí a doputují zpět (angl. *time-of-flight*, zkr. ToF), z čehož odvodí vzdálenost objektu v daném bodě od čidla. Je možné vyslat pulz a *time-of-flight* přímo měřit (*direct ToF*) nebo vysílat periodický signál a *time-of-flight* odvodit od fázového posunu vyslaného a přijatého průběhu (*indirect ToF*). Princip obou metod ilustruje obrázek č. 1. Přesnost určení vzdálenosti se odvíjí od přesnosti měření dané doby. V případě prvního přístupu je tedy potřeba extrémně přesné měření času nebo použití senzoru tohoto typu pro určování větších vzdáleností s menším rozlišením. V druhém případě má vliv zvolená perioda signálu, odvíjí se od ní zejména rozsah rozpoznávaných vzdáleností. K zamezení nejednoznačností je možno použít současně i více různých signálů s odlišnou periodou. Problémem

jsou v obou případech odrazy, kvůli nimž může jeden vyslaný signál dorazit vícekrát, v různou dobu, neboť putoval různými cestami [2].



Obrázek 1: Schema principu direct a indirect ToF [1]

Princip *indirect ToF* je použit např. u získávání tzv. „hloubkové mapy“ druhou generací senzoru Kinect.

2.2.6 Akustické snímání

Princip akustického snímání je velmi podobný předchozímu přístupu, místo elektromagnetického vlnění je však vysílán zvuk. Z toho vyplývá teoreticky vyšší dosažitelná přesnost určení vzdálenosti při stejné přesnosti měření času, zároveň však obecně nižší dosah. Komplikace s odrazy mohou být potenciálně řešeny snáze, neboť se zvuk pohybuje podstatně pomaleji a lze tedy lépe odlišit zaznamenání prvního odrazu od dalších [2]. Před následujícím vysláním totožného signálu je však nutné počkat pro vyloučení opožděného přijetí prvního signálu, čímž se snižuje využitelná snímací frekvence. Další nevýhodou je vliv teploty a vlhkosti vzduchu na rychlost šíření zvuku. Zejména v exteriérech se též projeví vliv směru a rychlosti větru. Metoda je navíc náchylná k chybám v důsledku zvukového šumu z okolí [2].

1 Obrázek odvozen z: http://campar.in.tum.de/twiki/pub/Chair/TeachingSs11Kinect/2011-DSensors_LabCourse_Kinect.pdf – slide 11 a 13

2.3 Možnosti kalibrace

Pojem kalibrace se může vztahovat ke dvěma souvisejícím problémům. Zaprvé jde o kalibraci vlastního senzoru za účelem kompenzace zkreslení získávaných dat, které je dáno jeho vnitřními parametry, zadruhé o jeho kalibraci v okolním prostředí, tedy o zjištění relace mezi souřadným prostorem dat ze senzoru a konkrétními body reálného prostoru. Uvedené metody jsou popisovány na příkladu zpracování obrazu (kalibrace kamery), jejich principy však lze v obecnější rovině uplatnit bez ohledu na formát dat.

2.3.1 Bez kalibrace

Jednou z možností, jak přistoupit ke kalibraci senzoru v kontextu reálného prostoru, je úplně ji vynechat. Veškeré body a pohyby jsou pak detekovány na základě jejich vlastní vzájemné polohy vzhledem k vhodně umístěnému senzoru. Předpokládá se, že senzor samotný (resp. jeho vnitřní parametry) je již dostatečně zkalibrovaný, např. přímo z výroby, úprava hodnot může probíhat v softwarové vrstvě použité k přístupu k datům ze senzoru. Toto řešení je vhodné pro gesta jako posun dolů či mávnutí doprava, pokud u nich nezáleží na tom, kde přesně v prostoru byla provedena. Takový přístup je uplatněn u většiny komerčních her využívajících pohybové senzory a lze se s ním setkat i při sestavování rozpoznávání gest za pomocí detekované kostry v prostředí Microsoft Kinect for Windows SDK.

2.3.2 Kalibrace kamery

Kalibrace vnitřních (angl. *intrinsic*) parametrů kamery se soustřeďuje na zjištění vlastností použité optické čočky zejména za účelem kompenzace optických vad (aberací) způsobených jejími nedokonalostmi, jako je zkreslení (distorze). Vnější (angl. *extrinsic*) parametry udávají souvislost souřadného prostoru kamery se sledovaným reálným prostorem, tedy natočení a posunutí kamery.

Existuje opět hned několik používaných přístupů ke kalibraci. Prvním extrémem je princip využívající dopřednou přesnou znalost kalibračních bodů, resp. jejich přesně daný posun, který je kamerou zaznamenáván. Druhý extrém tvoří kalibrace pomocí až předem nedefinovaného hýbání kamerou v nehybné scéně bez použití zvláštního kalibračního objektu (autokalibrace). Zhangova metoda tvoří kompromis mezi těmito přístupy [3]. Využívá vytištěný 2D šachovnicový vzor umístěný na pevnou podložku, jenž je snímán v obecně různých polohách vůči kameře. Je možno hýbat buď vzorem, nebo kamerou. Vzhledem k pravidelnosti vzoru, jehož podoba je předem dobře známa, a neměnnosti parametrů kamery během vlastní kalibrace, lze získat jak vnitřní, tak vnější kalibrační matici. Optické vady čočky jsou detekovány z narušení pravidelnosti vzoru, poloha a natočení kamery z pozice, velikosti a sklonu jednotlivých řad čtverečků.

Totožná či obdobná metoda kalibrace může být často integrována do různých knihoven a nástrojů pro počítačové vidění, např. Camera Calibration Toolbox for Matlab[2] (jehož implementace v jazyce C je součástí OpenCV), GML C++ Camera Calibration Toolbox[3] a další.

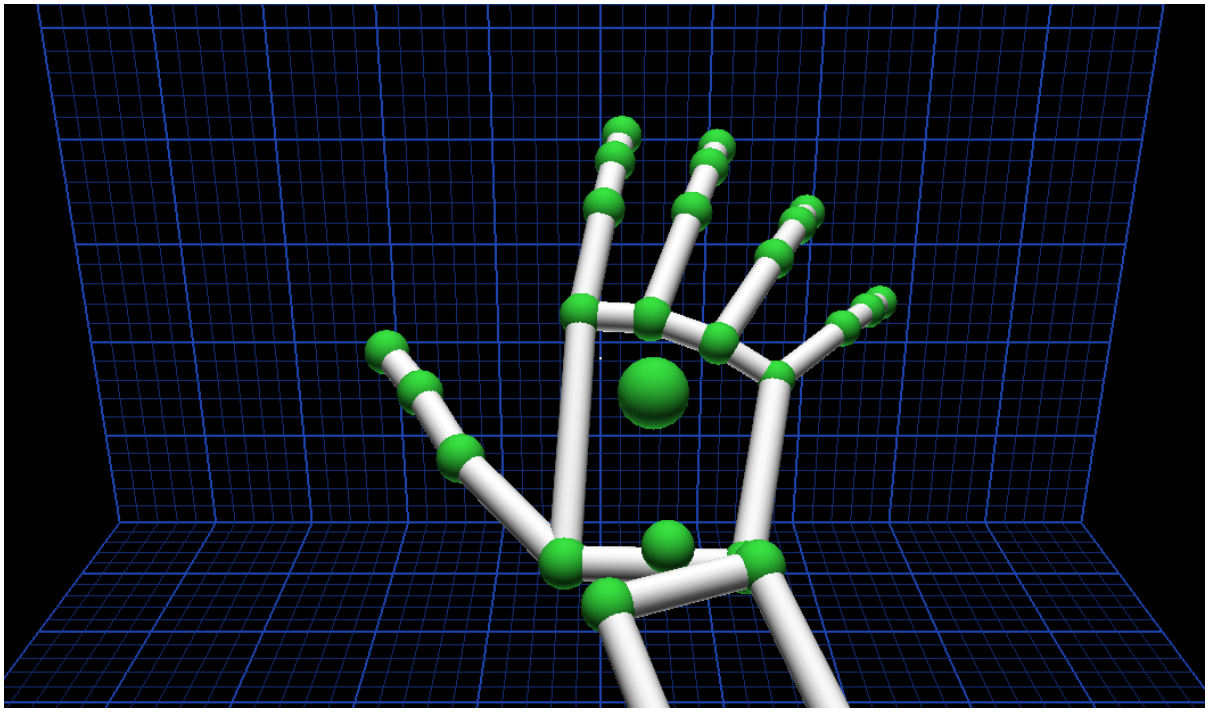
Je-li součástí zpracovávaných vstupních dat promítaný obraz, může být dále vhodné provést vzájemnou kalibraci projektoru a kamery. Tím lze získat souvislost polohy v rámci promítaného obrazu s polohou v obraze sledovaném, potažmo tedy poziční vztah mezi virtuální scénou a reálným světem, do nějž je promítána a ve kterém je s ní interagováno. Kalibrace projektoru se svým principem neliší od kalibrace kamery, obraz pravidelného vzoru je pouze promítán, místo aby byl zachycován, resp. je následně zachycován již zkalibrovanou kamerou.

2 Camera Calibration Toolkit: http://www.vision.caltech.edu/bouguetj/calib_doc/index.html

3 GML C++ Camera Calibration Toolbox: http://graphics.cs.msu.ru/en/node/909

2.4 Detekce otevřené ruky, prstů

Charakter dat ke zpracování pro ovládání programu gesty rukou závisí zejména na použitém senzoru. Může jít např. přímo o číselné hodnoty nějaké veličiny, rastrový obrázek, body v prostoru (*point cloud*) či již předzpracovaná data popisující kostru ruky s jednotlivými prsty a klouby.



Obrázek 2: Vizualizace kostry ruky získané pomocí senzoru Leap Motion [4]

V případě kostry (viz obrázek č. 2) je detekce ruky provedena přímo logikou senzoru, případně softwarovým rozhraním, každopádně mimo vlastní aplikaci.

V této kapitole budou uvažována data ve formátu obrazu (bitmapy), neboť v takové podobě je lze z mnoha senzorů získat a jsou v něm rovněž často používána v oblasti počítačového vidění, která s tématem úzce souvisí.

2.4.1 Segmentace

Data z libovolného senzoru je třeba nejprve rozdělit na pozadí a popředí, tedy oblasti, které by mohly představovat ruku. To se obvykle děje pomocí prahování, eventuálně s využitím nějakého pravděpodobnostního modelu [4].

4 Obrázek převzat z: http://cv.cs.nthu.edu.tw/upload/undergraduate/2015_article_3DDragon/index.html

Barevný obraz

Co se týče barevných obrazových dat, rozlišení oblasti potenciálně obsahující ruku od pozadí bývá provedeno na základě barvy, a častým problémem proto bývá vliv nasvícení na barvu kůže. Řešením může být kalibrace barev před použitím, spočívající například v umístění ruky uživatele na vyhrazené místo, navzorkování barvy v několika bodech a odvození přípustné barvy a tolerance odchylky od této hodnoty. V takovém případě však během používání aplikace nesmí docházet k výrazným změnám osvětlení prostředí, ve kterém má být ruka detekována.

Ke zmírnění vlivu vnějšího osvětlení může být využit barevný model HSL nebo podobný [5], za uplatnění pouze složek odstín (*hue*) a sytost (s*aturation*). Je to proto, že tento přístup, na rozdíl od jindy široce používaného modelu RGB, není citlivý na intenzitu jasu bílé složky osvětlení.

Způsobem jak zvýšit kontrast pozadí a ruky a tím podpořit úspěšnou detekci je použití rukavice neobvyklé a dobře detekovatelné barvy. Toto řešení však jde mírně proti snaze o přirozený způsob ovládání, neboť zavádí nutnost umělé pomůcky.

Pokud by bylo použito promítaného rozhraní tak, že bude promítáno přes ruku, která má být detekována, nabízí se teoretická možnost hledat rozdíly s obrazem z kamery s využitím znalosti podoby promítaného obrazu. Nelze však obrazy přímo odečíst. Bylo by nutné zohlednit posuny, rotace a změny měřítka vyplývající z toho, že pozice kamery a projektoru není stejná a z jejich polohy vůči pozadí, na které je promítáno. Kromě toho má vliv barevná charakteristika projektoru i kamery. Dále by bylo nutné vzít v potaz barevné vlastnosti pozadí a osvětlení nepocházející z projektoru.

Obraz z infračervené kamery

Použití infrakamery se zdrojem infračerveného světla, které bývá mj. používáno i jako tzv. noční vidění, minimalizuje vliv vnějšího osvětlení včetně promítání, pozadí by však stejně jako v případě barevných dat mělo zajišťovat co největší kontrast s rukou, tedy např. co nejméně odrážet světlo v daném spektru.

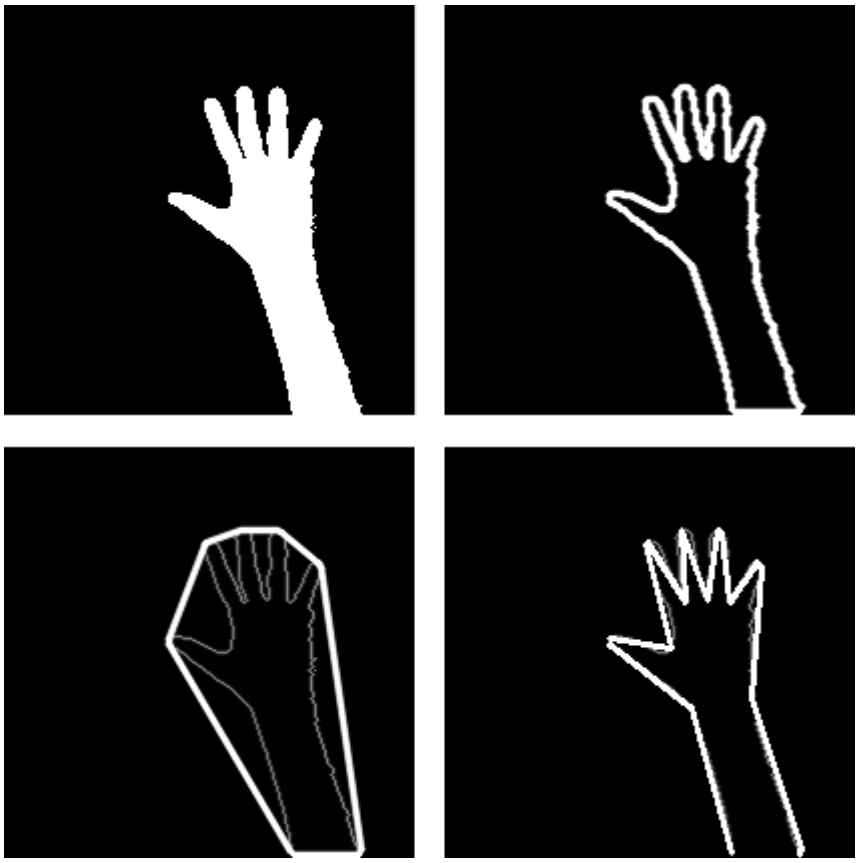
Hloubková data

Jsou-li k dispozici data popisující prostor, např. hloubku (vzdálenost od senzoru) v jednotlivých bodech, je nutné je rozdělit na část, kde se ruka vyskytuje, a zbytek. U zařízení typu tabletop bývá zpravidla touto hranicí vzdálenost desky stolu od senzoru.

2.4.2 Detekce prstů

Po oddělení popředí od pozadí je možné z dalšího zpracování vyloučit oblasti popředí, které jsou příliš malé, než aby obsahovaly otevřenou ruku, nebo alternativně uvažovat pouze největší takovou oblast, obzvláště pokud lze předpokládat, že se v záběru nevyskytují jiné objekty než pozadí a ruka.

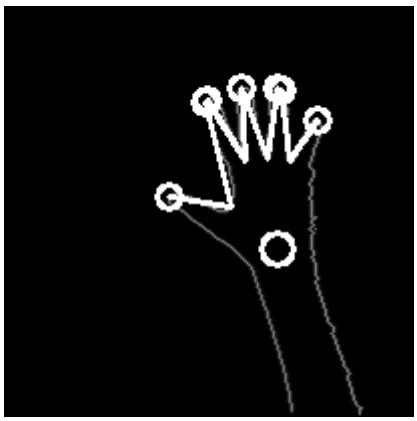
Dalším krokem je získání obrysu oblasti, její konvexní obálky a defektů v konvexitě [4]. Tento postup ilustruje následující obrázek. Pro jeho realizaci lze využít některou z knihoven pro zpracování obrazu nebo počítačové vidění, např. OpenCV[5].



Obrázek 3: Prahování, získání obrysu, konvexní obálky a defektů

Konvexní defekt je jev, kdy se obrys odchyluje od své konvexní obálky. Je charakterizován třemi body: začátkem, koncem a bodem obrysu, který je nejvzdálenější obálce, tedy nejhlubším místem defektu. Na obrázku č. 3 jsou defekty znázorněny pomocí spojnic bodů začátku s nejhlubším bodem a dále tohoto bodu s koncovým. Porovnáním vhodně zvolených podmínek pro vzájemnou polohu bodů můžeme rozhodnout, zda defekt odpovídá prostoru mezi prsty na otevřené ruce. Alternativně či ke zpřesnění výsledků lze využít informace o zakřivení (angl. *curvature*) špiček prstů. Počet prstů je o jedna větší počtu takových vad v konvexnosti. Body začátků, resp. konců konvexních defektů přibližně odpovídají pozici špiček prstů. Při nalezení dostatečného množství oddělených prstů lze objekt považovat za otevřenou ruku. Její pozice pak může být aproximována například těžištěm dané oblasti, jako tomu je na obrázku č. 4.

5 OpenCV FindContours, ConvexHull, ConvexityDefects: http://docs.opencv.org/2.4/modules/imgproc/doc/structural_analysis_and_shape_descriptors.html



Obrázek 4: Výběr „meziprstových“ defektů, označení pozic prstů a ruky

Z dostupných řešení pro detekci ruky lze jmenovat OpenNI 3D Hand Tracking Library[6] a Makematics FingerTracker[7].

2.5 Rozpoznávání gest

Jsou-li k dispozici informace popisující ruku, jako např. její pozici, orientaci a počet ukazovaných prstů, lze podle nich nebo na základě jejich časové souvislosti rozpoznávat jednotlivá gesta. Rozlišujeme gesta statická a dynamická. Statická nezahrnují pohyb, pouze přítomnost určitého postavení, tvaru ruky v obraze (např. znamení „OK“, vztyčený palec apod.), lze je tedy detekovat jen na základě tvaru ruky, např. podle momentů. Do této kategorie by bylo možné zahrnout i počítání ukazovaných prstů zmíněné v předchozí kapitole, pokud by bylo využito jako samostatný vstup programu. Další část textu se zabývá rozpoznáváním dynamických, pohybových gest, pro něž existuje několik různých metod výrazně se lišících svojí komplexitou. Zde jsou stručně popsány tři příklady.

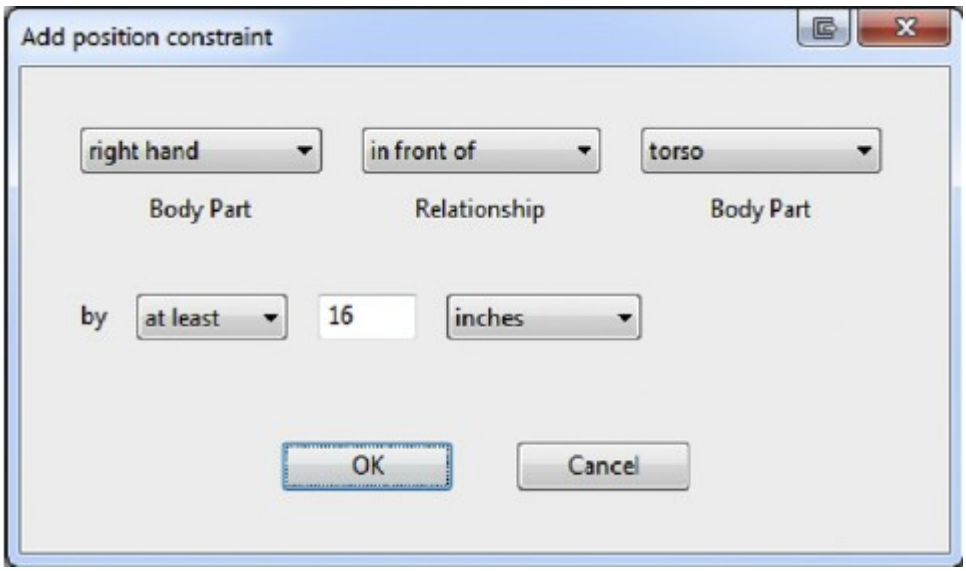
2.5.1 Analytický přístup

Pro jednodušší gesta lze zvolit přístup založený na rozdělení pohybu na jednotlivé fáze provádění a popisu podmínek, které musejí pro tyto fáze platit. Implementace spočívá v zápisu fází jako stavů konečného automatu s případnou tolerancí určitých odchylek v rámci sledování splnění daných podmínek. K těmto odchylkám může docházet např. vinou nepřesnosti a nekonzistence lidských pohybů nebo vlivem šumu v datech ze senzoru, je proto vhodné je rozumně zohlednit. S takovým

6 OpenNI 3D Hand Tracking Library: http://openni.ru/files/3d-hand-tracking-library/index.html

7 Makematics FingerTracker: http://makematics.com/code/FingerTracker/

způsobem specifikace gest se lze setkat např. v prostředí Microsoft Kinect for Windows SDK[8] nebo Flexible Action and Articulated Skeleton Toolkit (FAAST)[9].



Obrázek 5: Zadávání podmínek v GUI FAAST [10]

Pro složitá gesta s obtížně rozlišitelnými či popsatelnými fázemi, která mohou navíc být prováděna např. ve větším rozsahu přípustných rychlostí nebo mohou být různě orientována, může být tento popis příliš rozsáhlý anebo nedostatečně robustní, a je tedy vhodné zvolit některý ze sofistikovanějších přístupů.

2.5.2 Skryté Markovovy modely

Skryté Markovovy modely (angl. *hidden Markov models*, dále zkr. HMM) byly původně využity při zpracování řeči, lze je však použít v řadě dalších rozpoznávacích úloh. Vycházejí z Markovových řetězců, což je v podstatě stochastické (pravděpodobnostní) rozšíření konečných automatů, kde lze přechodům mezi stavy určit pravděpodobnost jejich provedení. U HMM není pozorovatelný

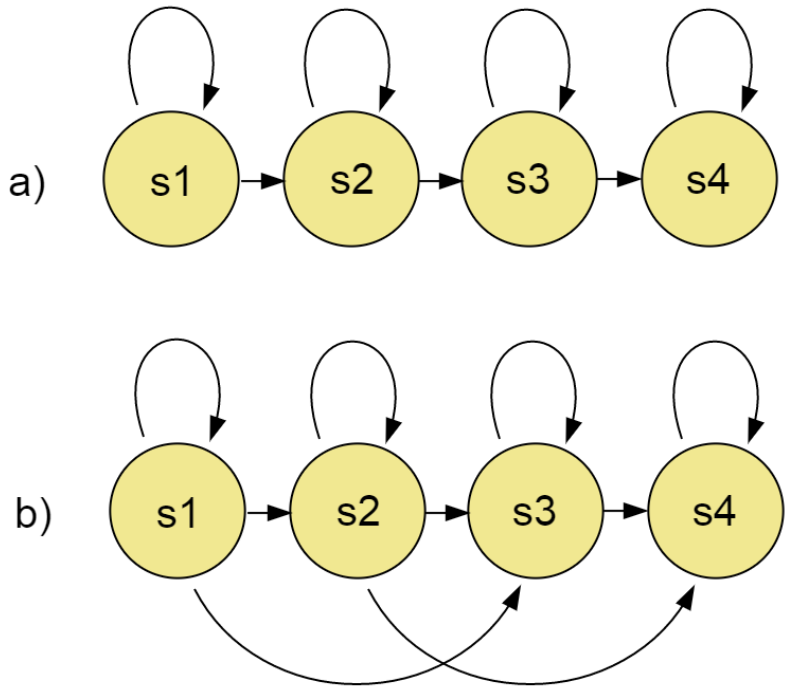
8 Specifikace gest v MS Kinect For Windows SDK: https://blogs.msdn.microsoft.com/mcsuksoldev/2011/08/08/writing-a-gesture-service-with-the-kinect-for-windows-sdk/

9 FAAST: http://projects.ict.usc.edu/mxr/faast/

10 Obrázek převzat z:
SUMA, Evan A., David M. KRUM, Belinda LANGE, Sebastian KOENIG, Albert RIZZO a Mark BOLAS. Adapting user interfaces for gestural interaction with the flexible action and articulated skeleton toolkit.*Computers & Graphics* [online]. 2013, **37**(3), 193-201 [cit. 2016-05-13]. DOI: 10.1016/j.cag.2012.11.004. ISSN 00978493. Dostupné z: http://linkinghub.elsevier.com/retrieve/pii/S0097849312001756

konkrétní průchod, tedy nelze určit přesný stav, v němž se HMM nachází, proto *skrytý*. Lze sledovat pouze výstupy, které jsou průchodem generované [6].

Postup může být následující: Pro každé gesto je vytvořen jeden model. Body v prostoru, kterými se během vykonávání gesta prochází, jsou rozděleny do několika shluků (angl. *cluster*). Každý stav modelu pak značí, že pozice ruky v daném časovém okamžiku přísluší ke konkrétnímu shluku. Dále je nastavena matice přechodových pravděpodobností, která pro jednotlivé modely specifikuje, s jakou pravděpodobností přejdou z jednoho stavu do jiného či setrvají v aktuálním, tedy odkud kam se může ruka při určitém gestu pohybovat. Může být umožněn přechod (nenulová pravděpodobnost) pouze do sousedního stavu a pouze jedním směrem (*left-right banded*, zkr. LRB, viz obr. 6a), přeskakování stavů (*left-right*, zkr. LR, viz obr. 6b) nebo může být uplatněna jakákoliv jiná, obecná topologie modelu [5]. Sledováním pozice ruky dostaneme posloupnost odpovídající možným stavům. Rozpoznané gesto je to, jehož model generuje takovou sekvenci s nejvyšší pravděpodobností [7].



Obrázek 6: Schematické znázornění LRB (a) a LR (b) topologie

K vylepšení matice pravděpodobností přechodů může být použito tzv. „učení s učitelem“ (angl. *supervised learning*). Existuje mnoho specifických typů HMM, které však nebudou dále popisovány. Samostatnou problematikou pak může být oddělení gesta (nalezení jeho začátku a konce)

v souvislých datech, k čemuž může být použit zvláštní model [8]. Přístup s využitím HMM je aplikován v Gesture Recognition Toolkit (GRT)[11].

2.5.3 Dynamické borcení času

Metoda dynamického borcení času (angl. *dynamic time warping*, dále zkr. DTW) má své kořeny také v klasifikaci řečových signálů. Jde v podstatě o zjištění podobnosti dvou signálů, jejichž části jsou mezi sebou různě posunuty nebo vůči sobě zrychleny či zpomaleny, tedy roztaženy či smrštěny. Spočívá v nalezení nelineární deformace časové osy, která vede k největšímu vzájemném přizpůsobení časových průběhů signálů, k jejich nejvyšší možné shodě [9]. Celý postup lze provádět i nad více dimenzemi zároveň (angl. *multi-dimensional DTW*, MD-DTW) [10]. Předmětnými signály mohou být v případě rozpoznávání gest např. časové průběhy pozice ruky v prostoru nebo přímo data z akcelerometrů v jednotlivých osách.

DTW je jednou z metod, kterou využívá Gesture Recognition Toolkit (GRT).

11 GRT: http://nickgillian.com/grt/index.html

3 Analýza a návrh

V této kapitole bude předložen návrh konkrétní podoby aplikace, která je předmětem řešení práce. Je zde mj. zdůvodněn výběr konkrétní karetní hry, popsána gesta pro její ovládání a nastíněny základní vlastnosti grafické stránky aplikace.

3.1 Výběr karetní hry

Jelikož má vytvořený program umožňovat uživateli hrát karetní hru, bylo nutné nějakou vybrat či vymyslet. Z principu použití promítaného rozhraní je automaticky problematický výběr jakékoliv hry, která spočívá v soutěžení hráčů, z nichž každý má své vlastní karty, utajené před dalšími hráči, neboť vyvstává otázka, jak tyto karty jednomu hráči zobrazit a zároveň je před ostatními skrýt. Bylo by to možné za použití oddělených promítacích ploch, překážky či časových fází, kdy jeden účastník hraje a ostatní jsou otočeni, a podobně. Takováto řešení však nejsou v souladu s obecným smyslem promítaného uživatelského rozhraní, totiž se snahou přiblížit vjem uživatele reálnému prostředí, proto se jejich volba k tomuto účelu nejeví jako příliš smysluplná a všechny takové hry byly předem zavrženy. Nabízí se tedy vybrat takovou hru, kde všechny karty jsou odkryté všem hráčům, případně se hry účastní jen jeden lidský hráč. Za soupeře mu eventuálně může posloužit počítač, komplexita rozhodování by ale spíše neměla být velká, aby těžiště této práce nebylo přesunuto z oblasti uživatelských rozhraní do problematiky umělé inteligence.

Z takto formulovaných možností byla vybrána hra blackjack. Všechny karty hráčů jsou v její variantě zvané *„face up“* všem odkryté, hráči nezávisle na sobě soupeří s krupiérem, jehož chování je jasně dané, přísně deterministické. Kromě toho je tato hra výhodná tím, že má zavedena gesta rukou, která jsou při jejím hraní používána. Tato gesta mají takovou podobu, že je lze při hře v kasinu sledovat kamerou shora (tzv. *eye in the sky*), mohou tedy být přejata i pro ovládání navrhované aplikace.

3.2 Ovládání

Před hrou samotnou se hráč přihlásí do hry tím, že vloží sázku. Jako u reálného blackjackového stolu bude i v aplikaci stanoven spodní a horní limit sázek. V rámci něj bude hráč volit umístěním otevřené ruky v příslušné výšce nad deskou stolu; čím vyšší poloha, tím vyšší sázka. Vložení sázky a zapojení se do hry poté potvrdí zavřením ruky a jejím krátkým setrváním na místě.

V závislosti na aktuální kombinaci karet a svých předchozích tazích bude mít hráč během hry na výběr až ze čtyř akcí. Standardně se označují anglickými výrazy *hit, stand, double down* (či jen

double) a *split*, a jejich význam a podmínky, které se na jejich uplatnění vztahují, jsou blíže uvedeny v příloze věnující se implementovaným pravidlům (Příloha 1). Zde jsou popsána gesta pro jejich vyvolání, která jsou používána v reálné hře:

- *hit* – poklepání prstem či prsty na stůl, přejetí prsty po stole směrem k hráči nebo mávnutí k sobě,
- *stand* – horizontální pohyb otevřené ruky do strany,
- *double down* – ukázání jedním prstem a zdvojnásobení sázky,
- *split* – ukázání dvěma prsty roztaženými do tvaru písmene V na aktuální kombinaci karet a vložením další sázky.

V souladu se snahou o maximální podobnost užívání navrhovaného programu s reálnou hrou blackjacku budou gesta rozpoznávaná aplikací obdobná. Rozdílem bude, že sázka se při využití možností *split* a *double* vloží automaticky, hráč pro to nic nedělá. Gesta pro tyto akce navíc bývají v opravdových hrách používána zejména v situaci, kdy není možné z pouhého umístění sázky rozhodnout, o kterou z nich má hráč zájem. V implementované aplikaci však bude nutné gesta provést vždy. Při použití gesta pro *double* v reálné hře je navíc jeden prst typicky ukázán krupiérovi, jako počet karet, který chce hráč obdržet, směřuje tedy vzhůru. Ukázání jedním prstem na karty může mít v některých kasinech význam *hit*. Při ovládání programu bude naproti tomu pro *double* vyhrazeno ukázaní jedním prstem horizontálně směrem ke krupiérovi.

Přesná podoba gest pro ovládání aplikace bude lépe patrná z videa na přiloženém DVD (Příloha 2).

Při detekci ruky bude předpokládáno, že se při hře na stole nevyskytují žádné cizí předměty, nicméně přesto bude uplatněna snaha o určitou robustnost použitého řešení, aby se se situací rozumně vyrovnalo i při porušení této premisy.

3.3 Konstrukce se stolem

K získávání vstupních dat a promítání grafického výstupu shora na stůl bude použita stávající konstrukce „inteligentního stolu“ ARTable v laboratoři robotiky O104 Fakulty informačních technologií, jejíž podobu zachycuje obrázek č. 7. Ta je osazena senzorem Kinect for Windows v2 (neboli Kinect for Xbox One s adaptérem) umístěným poblíž širokoúhlého HD projektoru.

Obrázek 7: Konstrukce ARTable (pozn.: Robot PR2 není k předmětu práce relevantní.)

3.4 Grafický návrh herní scény

Grafická část uživatelského rozhraní hry bude mít podobu kasinového stolu na blackjack s pozicemi pro čtyři hráče. Pro skutečný stůl na blackjack je obvyklý počet 5–7 hráčských pozic, vzhledem k velikosti stolu a za účelem omezení překrývání oblastí pro detekce gest hráčů při zachování jejich dostatečné velikosti byl však zvolen tento ne zcela standardní počet. Kolem každé takové pozice bude umístěna kruhová či elipsová úseč, která bude svou barvou reagovat na fázi hry a tedy upozorňovat hráče na určité situace, vyzývat jej k provedení gesta apod. Poblíž bude uveden součet hodnot hráčových karet a případně odpočet ukazující zbývající čas pro provedení akce. Tyto prvky kompenzují absenci fyzického krupiéra. Další grafické prvky nevyskytující se na reálném stole (obrázek č. 8) budou zcela omezeny. Po stole se budou pohybovat hrací karty a žetony. Celá scéna bude zpracována ve 3D.

Obrázek 8: Reálný stůl na blackjack při hře z pohledu hráče [12]

12 Obrázek převzat z: http://kstreetmagazine.com/wp-content/uploads/2013/04/casinogames.jpg

4 Implementace

Tato kapitola popisuje praktickou podobu aplikace ovládané gesty a využívající promítané uživatelské rozhraní, která umožňuje hrát karetní hru blackjack. Je zde zvoleno a popsáno vhodné vývojové prostředí a stručně vysvětleny některé aspekty architektury programu i konkrétní řešení vybraných dílčích problémů.

4.1 Výběr programového prostředí

Z plejády použitelných kombinací grafických a matematických knihoven, frameworků a herních enginů byla vybrána varianta Unity a Emgu CV. Hlavními argumenty ve prospěch Unity jsou kompatibilita se senzorem Kinect skrze wrapper, vestavěný systém synchronizace chování herních objektů s možností využití korutin, jejich začleňování do hierarchické struktury, možnost přehledné správy herní scény prostřednictvím editoru, dostupnost mnoha praktických knihovních funkcí, ať už jde o standardní knihovny jazyka C#, součásti enginu samotného nebo knihovny externí (příkladem může být třída pro práci s vektory). Implementačním jazykem bude zmíněný C#, a to proto, že je to jeden z programovacích jazyků použitelných v prostředí Unity a zároveň v něm lze nepřímo používat knihovnu OpenCV, která obsahuje množství optimalizovaných algoritmů využitelných zejména pro práci s obrazem za účelem rozpoznávání objektů. Použití OpenCV je možné prostřednictvím wrapperu Emgu CV, který poskytuje rozhraní pro její funkce jazykům softwarové platformy .NET.

4.2 Zprovoznění programového prostředí

Během uvádění zvolené kombinace programových prostředí do provozu se mohou vyskytnout určité potíže. Zde je stručně naznačen ověřený způsob umožňující propojit potřebné knihovny a dospět do funkčního stavu. Konkrétně je při implementaci použita toho času aktuální verze Unity 5.2 Personal Edition.

4.2.1 Emgu CV

Při rozpoznávání ruky je velmi výhodné využít možností knihovny pro zpracování obrazu, jako je OpenCV. Ta je z jazyka C# přístupná skrze výše zmíněný wrapper Emgu CV. Byla zvolena verze Emgu CV 2.4.9, o níž bylo předem zjištěno, že lze s Unity propojit. Je k tomu však potřeba provést několik úkonů:

1. Vytvořit v projektu složku `Assets/Plugins`.
2. Po instalaci uvedené verze Emgu CV nalézt vytvořenou složku `bin`.
3. Překopírovat všechny obsažené soubory s příponou .dll (celkem 12) do složky `Plugins`.
4. Obdobný postup i pro soubory ve složce `bin/x64` (celkem 22).
5. Soubory `cudart64_50_35.dll` a `npp64_50_35.dll` dále nakopírovat i do složky s instalací Unity Editoru (výchozí je `Program Files/Unity/Editor`).
6. Nastavit v *Build Settings* projektu v rámci Unity Editoru *Architecture* na „x86_64“.

I při dodržení tohoto postupu se mohou vyskytnout chybové hlášky, jako např. ta zachycená na obrázku č. 9. Řešením je postup opakovat nebo alternativně vyzkoušet soubory ze složky `bin/x86`.

Obrázek 9: Příklad chybové hlášky při integraci Emgu CV do Unity

Obdobný postup kopírování souborů .dll je nutno aplikovat i při zprovozňování aplikace vyexportované do spustitelného souboru.

Dále je důležité zprovoznit knihovnu `System.Drawing`, která zahrnuje třídy jako `Rectangle` a `Point`, jež jsou při práci s Emgu CV hojně využívány. K tomu je zapotřebí nalézt soubor dané knihovny v systému (`System.Drawing.dll`; typicky ve složce `Program Files/Unity/Editor/Data/Mono/lib/mono/2.0`) a tento zkopírovat do vytvořené složky `Plugins` v projektu. V položce *Player Settings* v *Build Settings* Unity Editoru je třeba nastavit *Api Compatibility Level* na hodnotu „.NET 2.0“ (nikoliv pouze *Subset*). K odstranění upozornění na

chyby ve vývojovém prostředí (tj. obvykle Visual Studio nebo MonoDevelop) je též nutné přidat referenci na předmětnou knihovnu (*Project → Add Reference…/Edit References…*).

4.2.2 Aplikační rozhraní Kinectu

Jak bylo zmíněno v předcházející kapitole věnované návrhu, použitým senzorem bude Kinect for Windows v2. Aby bylo možné s ním pracovat v prostředí Unity, je nutné buď použít nějakou vrstvu zajišťující vzájemnou kompatibilitu, nebo rozdělit aplikaci na klienta reprezentujícího hru v Unity a server zpracovávající data z Kinectu a zpřístupňující je klientovi.

Z existujících softwarových produktů slibujících zmíněnou funkcionalitu byl vybrán wrapper KinectForWindows UnityPro 2.0.1410[13]. Je původně určen pro verzi Unity Professional Edition, nicméně od Unity verze 5 lze do projektu přidávat externí pluginy i v edici Personal a lze jej tedy využít i tam.

Dostupný archiv obsahuje funkcionalitu standardního SDK rozdělenout do tří souborů ve formátu .unitypackage. Jde o plugin umožňující základní práci se senzorem, dále modul pro rozpoznávání obličejů a nakonec modul pro specifikaci vlastních gest. V rámci implementace aplikace byl použit pouze první z nich. Do projektu v Unity jej lze importovat přes nabídku *Assets → Import Package → Custom Package*.

V době vzniku práce jsou jedinými operačními systémy oficiálně podporovanými pro práci s Kinectem v2 skrze SDK systémy Windows 8, Windows 8.1 a Windows Embedded Standard 8 (ačkoliv existují i možnosti jak jej využívat na některých linuxových distribucích). Připojení Kinectu v2 k počítači je možné pouze skrze rozhraní USB 3.0 za použití speciálního adaptéru. Tyto informace jsou zde uvedeny zejména jako upozornění pro čtenáře, který by měl případný zájem zprovoznit analogické prostředí na vlastním počítači.

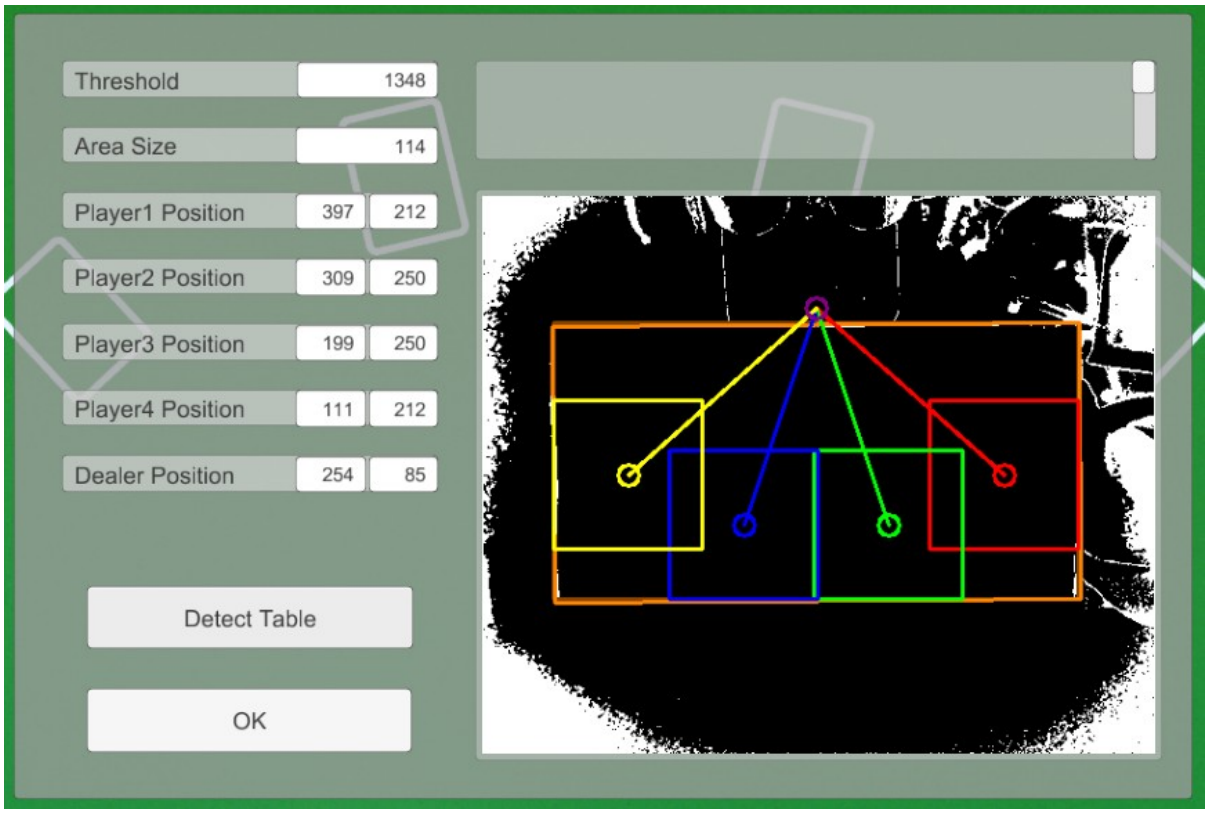
4.3 Kalibrace

Ovládání aplikace bylo navrženo tak, že není potřeba provádět „plnohodnotnou“ kalibraci senzoru s projektorem. Je nicméně vhodné ustanovit oblasti pro jednotlivé hráče, ve kterých budou detekována jejich gesta a zároveň zjistit vzdálenost stolu od senzoru. Za tímto účelem se program po spuštění pokusí naskenovat stůl. Předpokládá se přitom, že v bodě tvořícím střed záběru Kinectu se stůl nachází a že je plocha stolu přibližně kolmá na spojnici tohoto bodu se senzorem. Jinak může být stůl libovolně posunut a natočen. Použitý algoritmus skenování stolu je čtyřsměrové semínkové vyplňování (angl. *flood fill*) s tolerancí odchylky sousedních pixelů.

13 KinectForWindows wrapper: https://developer.microsoft.com/en-us/windows/kinect/tools (položka Unity Pro packages)

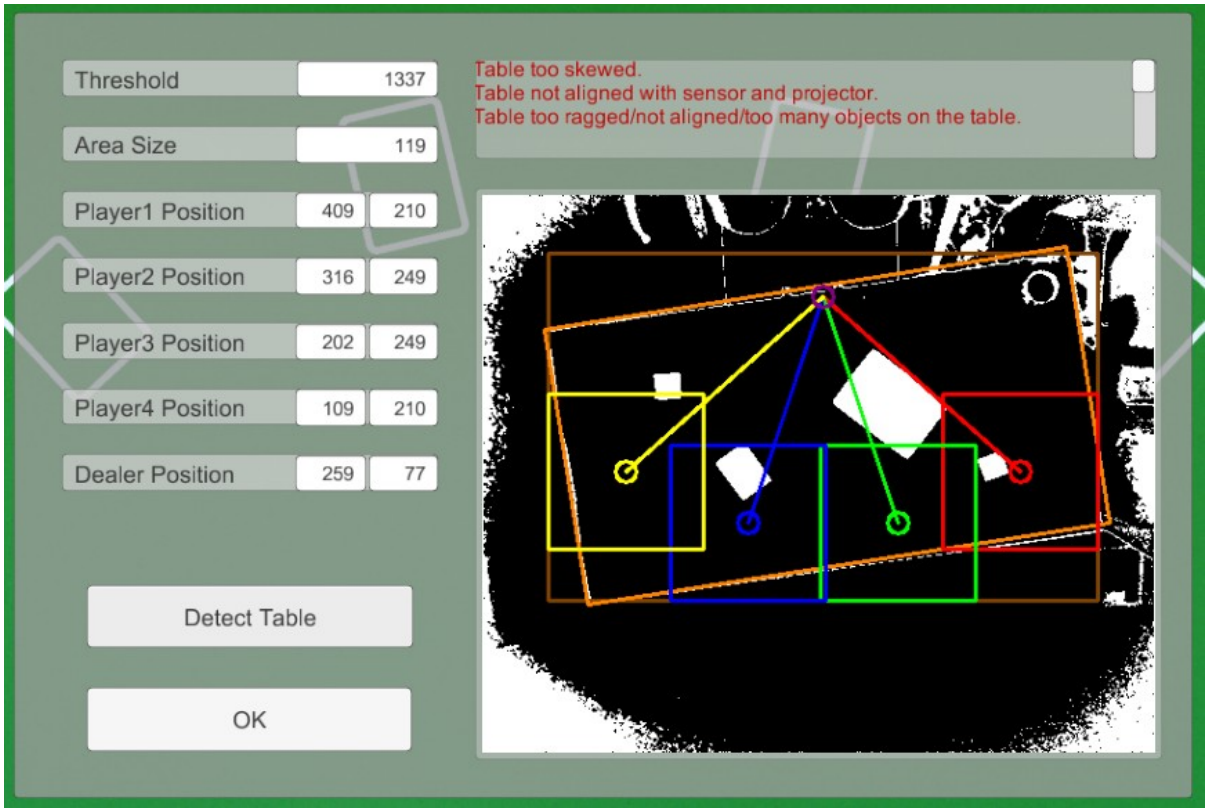
Plocha stolu je získána jako opsaný obdélník vyplněné oblasti. Na takto zjištěnou plochu jsou umístěny čtverce, které budou později použity jako oblasti pro rozpoznávání gest hráčů (tzv. *region of interest*, zkr. ROI). Při jejich rozvržení je brána v potaz pouze šířka stolu a jeho spodní okraj (ten, který je nejblíže hráčům a nejdále od konstrukce). Z toho vyplývá, že pro optimální zážitek ze hry je vhodné, aby byl obraz projektoru zarovnán s touto hranou stolu a měl odpovídající šířku, pro fungování aplikace jako takové to však není zcela zásadní. Další parametry vzájemného umístění stolu a projektoru nejsou pak vůbec podstatné. Kromě toho je určena pozice virtuálního krupiéra, která je jednotlivým hráčům referencí pro směr provádění jejich gest.

Poté, co proběhne detekce stolu a rozvržení oblastí, je zobrazeno formulářové rozhraní, ve kterém lze před spuštěním hry samotné za použití klávesnice manuálně upravit detekované hodnoty nebo s pomocí přítomného náhledu posunout fyzický stůl do korektní pozice (obrázek č. 10). Pokud chce uživatel ruční úpravu hodnot zrušit nebo zohlednit posunutí stolu v hodnotách, použije k tomu tlačítko „Detect Table“. Náhled je dostupný v podobě hloubkové mapy naprahované zadanou hodnotou s vyznačeným stolem, oblastmi pro hráče a pozicí krupiéra. První hráč sedí zcela vpravo, poslední zcela vlevo.



Obrázek 10: Rozhraní pro úpravu hodnot z detekce stolu při korektním postavení

Jsou též zobrazeny případné připomínky či doporučení týkající se polohy, velikosti, natočení a sklonu stolu, které by měly usnadnit dosažení správného nastavení konstrukce (obrázek č. 11 vpravo nahoře).



Obrázek 11: Rozhraní pro úpravu hodnot z detekce stolu při nevhodném umístění stolu vůči zbytku konstrukce

Na obrázku č. 11 je zřetelné otočení stolu (oranžová; hnědou barvu má opsaný obdélník) vůči senzoru a množství objektů, které na něm leží. Na to upozorňují nápisy v horní části okna.

Předpokládá se, že po spuštění hry se již vzájemné postavení stolu a zbytku konstrukce nijak výrazně nemění. Pokud by k tomu došlo, je nutné aplikaci restartovat.

4.4 Detekce ruky

Otevřená ruka je detekována, pokud se podařilo nalézt více než tři oddělené prsty. K tomu slouží přístup popsaný v kapitole 2.4.2. Použitou podmínkou pro rozdělení defektů je $SDE \leq 80$ °, kde S je začátek defektu, D je nejhlubší bod a E je konec defektu, přičemž nejsou uvažovány nesmyslně malé defekty (hloubka menší než 3 pixely).

Samostatným problémem je detekce jednoho prstu. Je-li přítomen právě jeden „meziprstový“ defekt, je počet prstů roven dvěma. Pokud nebyl nalezen žádný, může hráč ukazovat žádný či jeden

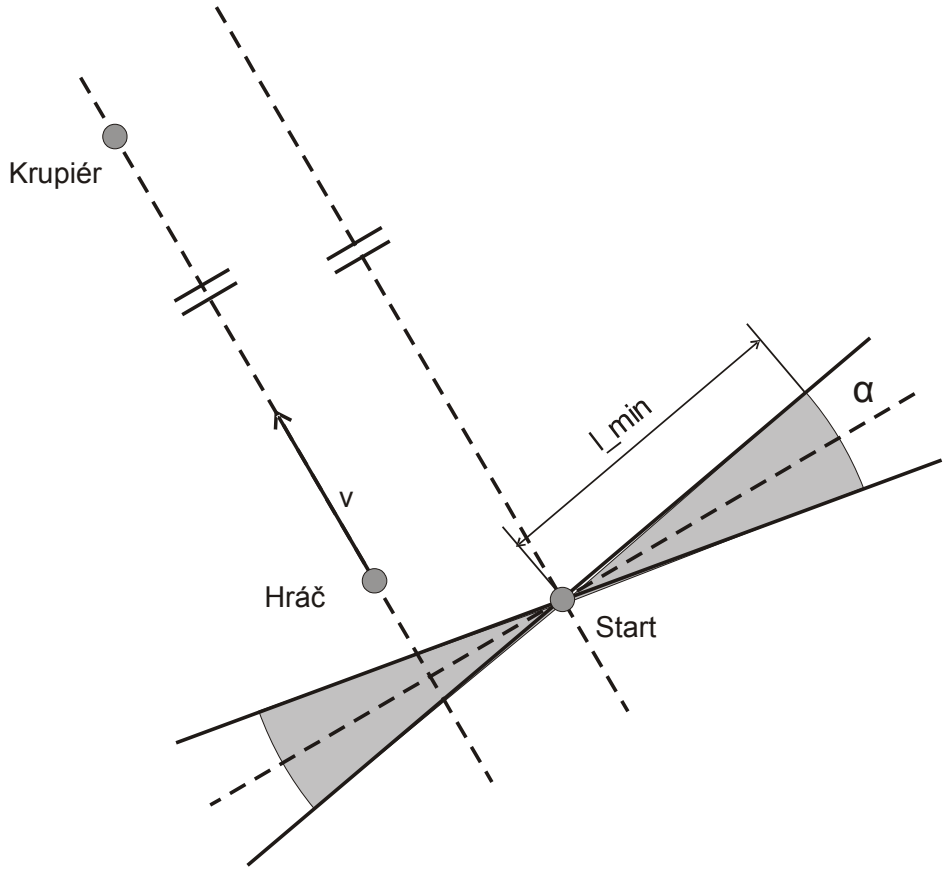
prst. V takovém případě je podmínka otočena. Nekontroluje se úhel mezi začátkem, nejhlubším bodem a koncem jednoho defektu, ale úhel mezi nejhlubším bodem jednoho defektu, koncem téhož/začátkem sousedního a nejhlubším bodem sousedního defektu. Jeho velikost nesmí přesáhnout 30 °. Zároveň je vyžadovaná určitá minimální délka jednoho prstu. Komplikací je to, že jeden koneček prstu může být rozdělen na několik konvexních defektů. Ze zpracování je tedy nutno vyloučit (přeskočit) příliš malé defekty. Tím však vyvstává nutnost kontroly vzdálenosti konce jednoho a začátku dalšího defektu, neboť mezi nimi se mohlo nacházet větší množství drobných defektů, který byly vynechány. Tato vzdálenost musí být menší než maximální přípustná šířka jednoho konečku prstu. Jako středový bod ověřovaného úhlu je použit střed spojnice konce a začátku uvažovaných defektů.

4.5 Rozpoznávání gest

Gesta jsou popsána analytickým způsobem. K toleranci odchylek od očekávané podoby gest je využit plovoucí čítač s omezením maxima, který je inkrementován v případě splnění podmínek a dekrementován při jejich nesplnění. Klesne-li jeho hodnota na nulu, rozpoznávání gesta je resetováno, např. postup časovače pro sledování délky ukazování je anulován, pozice začátku gesta pro posun je rovna aktuální pozici ruky. Zároveň je použit časový odpočet pro případ, že by posun trval příliš dlouho (pohyb ruky byl příliš pomalý). Pokud bylo před jeho vypršením gesto provedeno v dostatečném časovém či prostorovém rozsahu, je vyvolána související akce.

Předpokládá se, že hráči u stolu nesedí přímo, čelem kolmo na nejbližší hranu stolu, ale naopak jsou rozesazeni podél myšlené kruhově zahnuté hrany, jako u opravdového stolu na blackjack. Je tudíž nutné rozpoznávaný směr provádění jejich gest přizpůsobit tomuto natočení. K tomu účelu jsou použity 2D vektory popisující směr od hráče ke krupiérovi při pohledu shora. Veškeré pohyby rukou jsou vyjádřeny také vektorem, získaným odečtením aktuální pozice ruky od pozice dřívější. Směr pohybu je dán úhlem mezi těmito dvěma vektory.

Následující obrázek ilustruje popsaný princip v kontextu gesta pro akci *stand*, tedy posunutí otevřené ruky do levé či pravé strany.



Obrázek 12: Využití vektorů pro určení směru gesta

Body *Hráč* a *Krupiér* označují pozici daných aktérů. Vektor *v* určuje směr od hráče ke krupiérovi. V bodě *Start* bylo zahájeno provádění gesta. Gesto je nutné vykonat minimálně v rozsahu *l_min* doprava či doleva od tohoto bodu. Úhel *α* udává rozsah přípustné odchylky od ideálního směru provádění (tolerance je $\pm\frac{\alpha}{2}$). Tento ideální směr je vztažen právě k vektoru *v*. Zjednodušeně řečeno, pokud se tedy pohyb ruky hráče odehrává ve vyplněné oblasti a dosáhne vzdálenosti *l_min* od *Start* před vypršením časového limitu, je vyvolána příslušná akce, v tomto případě *stand*. Všechna ukazovací a posunová gesta využívají tyto vektory obdobným způsobem.

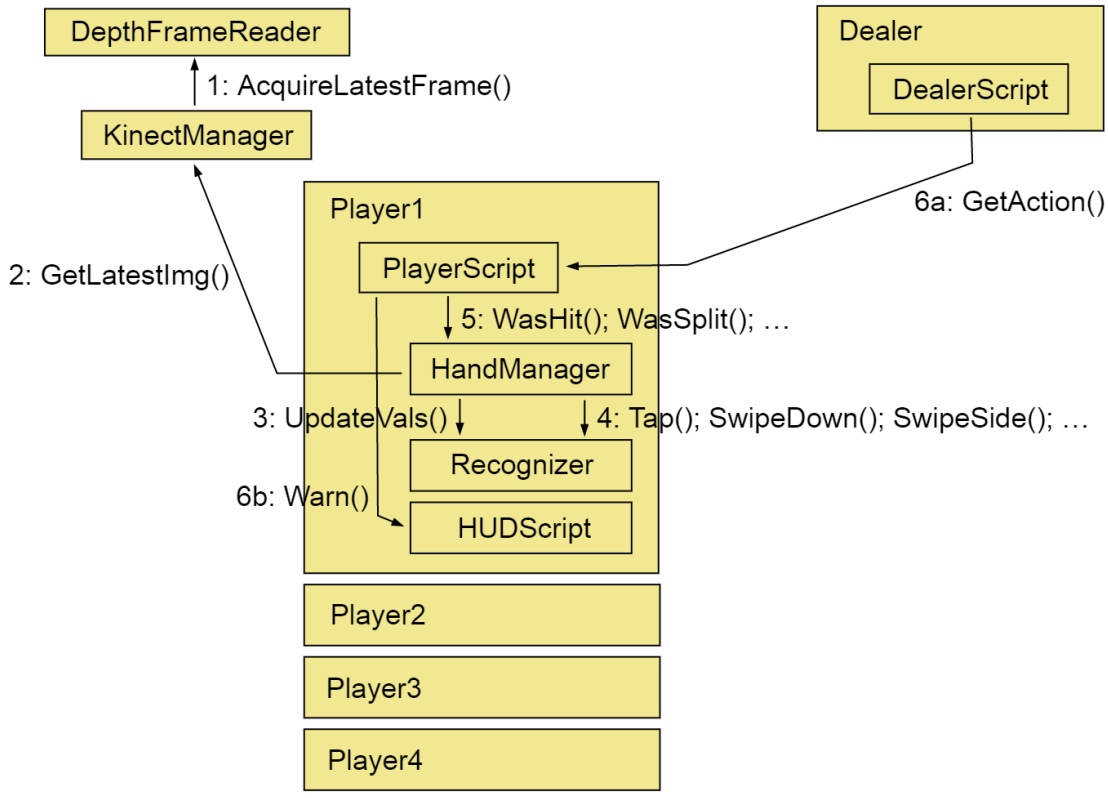
Směr od hráče ke krupiérovi je dán pozicí středu oblasti pro rozpoznávání gest daného hráče a pozicí krupiéra, přičemž obě tyto hodnoty jsou získány v rámci detekce stolu a lze je manuálně upravit v korekčním GUI zmíněném výše.

4.6 Struktura aplikace

Scéna v Unity se skládá z jednotlivých herních objektů (*GameObjects*) uspořádaných do hierarchické struktury. Objekty obsahují určité komponenty. Z hlediska vlastní programové implementace je zajímavá komponenta *Script*. Tou lze obstarat jak backend (např. logika chování objektů, pravidla hry), tak frontend (např. animace pohybů, barvy) aplikace. Tato kapitola stručně představuje vybrané aspekty implementace, zejména co se týče vytvořených tříd, jejich funkce a jejich vzájemného provázání.

4.6.1 Ovládání

Obrázek č. 13 v krocích 1 a 2 schematicky znázorňuje předávání hloubkové mapy ze senzoru Kinect. `KinectManager` čte data, provádí prahování na 16 bitech, převádí vzniklý obrázek do osmibitové formy, kterou vyžadují některé dále volané funkce Emgu CV. `HandManager`, který je hierarchicky potomkem každého objektu hráče, takto zpracovaná data z `KinectManageru` načítá. To je synchronizováno buď frameratem hry, nebo dostupností nového snímku z Kinectu podle toho, co je pomalejší, více omezující. V optimálním případě se tak děje třicetkrát za jednu vteřinu (maximum Kinectu).



Obrázek 13: Diagram komunikace objektů

Obrázek č. 13 dále ilustruje postup při detekci gesta a vyvolání související akce. `HandManager` v rámci příslušného ROI hráče (získané ve fázi detekce stolu) hledá ruku a její parametry (pozici, počet ukázaných prstů, směr ukazování při detekci jednoho nebo dvou prstů apod.). Poté volá `Recognizer` (3), který na základě těchto aktuálních hodnot provede přechod příslušných konečných automatů pro rozpoznávání gest do nového stavu. Následně zjišťuje provedení jednotlivých gest (4). Došlo-li k úspěšnému rozpoznání gesta, `HandManager` mu přiřadí akci (např. poklepání a posun směrem k hráči jsou dvě různá gesta pro tutéž akci). Tuto situaci zjišťuje `PlayerScript` (5) a poté ověří, zda je příslušná akce legální v kontextu dané herní situace. Buď připraví její předání krupiérovi (6a), nebo upozorní hráče na neplatnost tahu pomocí HUD (6b). Na informace o akci hráče se dotazuje krupiérův `DealerScript`, hráč odpovídá pomocí výčtového typu. Hráč, který právě není na tahu, je deaktivován, neprobíhá rozpoznávání jeho gest.

4.6.2 Animace

K provádění veškerých pohybů objektů je využit `OneMoveScript`, který je připojen jako komponenta pohybovaného objektu. Při vytvoření jsou mu předány parametry kýženého pohybu (cílová pozice a rotace, rychlost pohybu a její průběh). Podporován je konstantní pohyb a analogie pohybu po pružině (postupné zrychlení a zpomalení pohybu), k tomu je využito volání příslušné funkce Unity[14]. Při napojení nové instance `OneMoveScript` na příslušný objekt jsou zrušeny všechny další instance, které s objektem souvisí. Tento přístup nevyniká svou efektivitou, jde však o jednoduché a přehledné řešení, při jehož dodržení nedochází k vyvolávání několika rozporných pohybů, které vedou k potenciálně nekonečnému provádění (objekt nikdy nedoputuje do jejich cíle).

4.7 Grafika

Použité textury karet a žetonů byly převzaty[15, 16]. Jsou volně dostupné pro nekomerční použití. Textury žetonů byly mírně upraveny a aplikovány na vytvořené modely. Všechny další grafické prvky jsou původní. K jejich tvorbě byly použity programy Blender, Paint.NET a Inkscape.

14 UnityEngine.Vector3.SmoothDamp: http://docs.unity3d.com/ScriptReference/Vector3.SmoothDamp.html

15 Textury hracích karet: http://wheels-cards.wc.lt/bicycle_cards.html

16 Žetony: http://sparky3d.net/?p=578

5 Testování

Testování proběhlo za přispění 4 studentů FIT VUT v Brně, při jejich výběru byly cíleně hledány osoby s různou velikostí rukou. Časový rozsah jednotlivých testů byl přizpůsoben konkrétním možnostem zúčastněných, v průměru strávila každá osoba hraním hry cca 2,5 hodiny. Aplikace byla spuštěna na notebooku Lenovo ThinkPad Edge E531 (Intel Core i3 2,5 Ghz, 4 GB, MS Windows 8.1 64bit). Všichni účastníci měli pouze zcela minimální či vůbec žádné předchozí zkušenosti s hrou blackjack. Před samotným testováním jim byla vysvětlena použitá pravidla se současným popisem a předvedením očekávané podoby gest, jimiž jsou iniciovány příslušné akce hráče. Po přípravě zařízení a spuštění aplikace následně proběhla vlastní hra, při níž byla sledována úspěšnost vyvolávání kýžených tahů jednotlivými testovacími osobami. V závislosti na dílčích výsledcích testování byla implementace ovládání pozměňována, objektivní měření tedy nejsou dobře srovnatelná a je nutné brát v potaz spíše subjektivní hodnocení účastníků testování.

Ve všech případech se během několika počátečních kol objevovalo značné množství neúspěšných pokusů. Obzvláště u gesta pro *double* a poklepání zpočátku počet neúspěšných pokusů vysoce přesahoval počet těch úspěšných. Poměr se během několika málo odehraných tahů výrazně zlepšoval. Pokud u stolu seděl hráč, který měl již s ovládáním zkušenosti, a snahu svých spoluhráčů korigoval, byl tento progres pozorovatelný rychleji. Většina hráčů postupem času volila pro akci *hit* gesto posunu k sobě namísto alternativního poklepání, které bylo méně spolehlivé. Úspěšnost rozpoznání poklepání je značně závislá na přesnosti manuální korekce hodnoty prahu získané během detekce stolu před započetím hry.

Po dostatečném množství pokusů o provedení gest již hráčům ovládání nečinilo žádné větší potíže. Osoby zúčastněné na první fázi testování se shodly na časovém údaji 15–30 minut jako na době, po které již byli s gesty uspokojivě sžiti, poslední účastník (po úpravách implementace) uvedl dobu 5 minut. Na základě tohoto samostatného údaje však nelze tvrdit, že iterativní úpravy vedly k podstatnému zlepšení ovládání, neboť daný parametr je značně subjektivní a počet vzorků je příliš malý, není tedy statisticky vypovídající.

Poté, co si však takto „zkušený“ hráč přesedl k jinému místu u stolu, měl často alespoň po dobu jedné hry s ovládáním opět zvýšené potíže. Tento fakt lze připsat na vrub skutečnosti, že obraz projektoru u použité konstrukce nezasahuje na celou šířku stolu, což byl jeden z předpokladů při návrhu uživatelského rozhraní. Tento předpoklad vychází myšlenky, že při použití aplikace se stolem obvyklých rozměrů bude uplatněna snaha využít celou plochu stolu, a bude tedy přirozeně žádoucí umístit projektor příslušným způsobem bez zvláštního ohledu na aplikaci samotnou. Pro toto vysvětlení hovoří to, že problémy byly výraznější v případě, kdy si hráč přesedl z místa naproti

krupiérovi ke krajní pozici. Předmětný stůl má totiž svou hloubku podstatně menší než šířku a projektor je umístěn tak, aby promítaný obraz nezasahoval mimo něj. Zmíněná konstrukce je však využívána i k jiným aktivitám, je jím přizpůsobena, a nezdálo se tedy vhodné měnit její dispozice. Popsaný efekt by mělo být možné kompenzovat ručním posunutím oblastí hráčů, které lze provést před zahájením hry, nepodařilo se však spolehlivě rozhodnout, vede-li to k lepším či horším výsledkům, neboť obtíže hráčů netrvaly dostatečně dlouho.

Jak již bylo předestřeno, během testování probíhaly úpravy implementace ovládání. Tyto spočívaly téměř výlučně ve zpřísňování podmínek pro rozpoznávání jednotlivých gest tak, aby v případě sporně provedeného pohybu bylo nutné gesto raději opakovat než vyvolat nesprávnou akci. Největší takový problém byl u gest pro *double* a poklepání, jež si jsou shora značně podobná a zároveň podmínky pro zahrání těchto akcí jsou dosti často splněny současně. Osoby, které se účastnily testování před i po takových úpravách, hodnotily zvolený přístup jednoznačně kladně.

Testováním bylo dále zjištěno, že je gesta vyžadující detekci otevřené ruky výhodnější provádět na pravé straně stolu levou rukou a naopak. Je to dáno přirozeným náklonem ruky a tím, že jde o bodové (perspektivní) snímání, nikoliv plošné (ortogonální), v důsledku čehož se mohou jednotlivé prsty v obraze za sebe zakrývat. Tento efekt je mimoděk zmírněn rozesazením hráčů po myšleném obvodu zaobleného stolu.

Během testování aplikace byl pořízen obrazový materiál, který se stal základem přiloženého instruktážně-propagačního videa. Bylo by vhodné provést rozsáhlejší testování aplikace, přičemž by toto video mohlo být použito jako součást průpravy testovacích osob.

6 Možnosti dalšího rozšíření

Účastníkům testování nejčastěji chyběla možnost zjistit, jak si ve hře vedou dlouhodobě. Implementace dlouhodobého skóre hráčů mezi jednotlivými hrami nepředstavuje z pohledu herní logiky výraznější problém, v současné podobě aplikace by však její nasazení nebylo zcela korektní, neboť neexistuje žádná možnost, jak určit, který hráč přišel či odešel, nebo kdo si kam mezi hrami přesedl. Tím pádem lze sledovat skóre konkrétní hráčské pozice, nikoliv však konkrétního hráče. Bylo by možné zavést gesto příchodu a odchodu hráče na/z dané pozice, toto řešení však postrádá určitou eleganci a do jisté míry narušuje přirozenost ovládání. Lepší by bylo využít toho, že Kinect v použité konstrukci snímá i nejbližší okolí stolu, a při prvním přihlášení hráče hledat např. jeho hlavu (v oblasti souvislé s příslušnou rukou by byl po postupném zvyšování úrovně prahu hledán tvar podobný kruhu). Sledováním výšky a vzdálenosti hlavy od stolu by bylo možné rozhodnout o okamžiku, kdy hráč opustil stůl a čekat na příchod hráče dalšího. Skóre by bylo hráči počítáno od jeho příchodu do odchodu.

Většinu lidské společnosti a tedy potenciálních uživatelů aplikace tvoří praváci, kteří výšku sázky určují častěji pravou rukou. V rámci testování vyšlo najevo, že při vkládání sázky si hráči pravou rukou často zakrývají nápis označující její hodnotu, protože tento nápis je umístěn vpravo od pozice hráče. Toto umístění vyplývá z faktu, že je využit stejný grafický prvek jako při vypisování součtu hodnot karet, a ten je umístěn vpravo proto, že při *splitu* je druhá sázka vložena vlevo tak, aby vizuálně korespondovala s druhou kombinací karet, jež je také vlevo. Řešením je použít samostatné pole pro zobrazení výšky sázky vlevo nebo úplně předělat celou konstelaci objektů v dané oblasti. Bylo by tak možné např. umisťovat druhé sázky po akcích *double* a *split* rozdílně a reflektovat tak formální náležitosti, které se vyskytují v reálné verzi hry (sázka pro *double* bývá obvykle umístěna do boxu, zatímco u *splitu* nikoliv), jež byly zjednodušeny, neboť z pohledu vlastní práce nejsou nijak zásadní.

Hra je zpracována ve 3D, proto by bylo možné využít natočení a posunutí virtuální scény tak, aby kompenzovala drobné nepřesnosti ve vzájemném umístění projektoru a stolu. Za tím účelem by však bylo nutné doplnit plnohodnotnou kalibraci celé soustavy senzoru a projektoru.

Nabízí se uplatnění dynamického odvozování počtu hráčských pozic z velikosti použitého stolu. Bylo by dále možné doplnit vizuální efekty, jako jsou částicové efekty, nápisy při jednotlivých situacích („win“, „bust“ apod.), vysvětlivky neplatných tahů a další. Vzhledem ke snaze maximalizovat podobnost aplikace s reálnou situací by však bylo ještě vhodnější tyto doplňkové informace sdělovat hráči zvukem, verbálně, pomocí předpřipravených slovních komentářů. Z pohledu

logiky hry by implementace nebyla složitá, neboť vnitřně jsou jednotlivé důvody pro neplatnost tahu rozlišovány a je s nimi pracováno s využitím výčtového typu.

Zároveň s použitím zvukového výstupu programu by bylo vhodné použít i zvukový vstup pro ovládání hry. Použití slovních pokynů jako alternativy gest by ještě zvýšilo přirozenost ovládání.

Stávající aplikace by měla být spolu s případnými rozšířeními dále testována. Měl by být rozšířen okruh testovacích uživatelů tak, aby zahrnoval zejména osoby různého věku a různé úrovně zkušeností ve hře blackjack. U jednotlivých gest by měla být zaznamenávána úspěšnost jejich provedení. Dále by bylo vhodné lépe otestovat odolnost aplikace vůči umísťování cizích předmětů na stůl. V neposlední řadě by mělo být vyzkoušeno její chování při použití stolů různých velikostí a různých vzdáleností od nich. Na základě takového testování by pak bylo možné odvodit vztahy pro podmínky jednotlivých gest a velikosti/vzdálenosti stolu, čímž by se aplikace stala flexibilnější, lépe použitelnou ve větším rozsahu reálných prostředí.

7 Závěr

Předmětem práce bylo vytvoření počítačové karetní hry s projekcí grafického výstupu na stůl a ovládáním pomocí gest. Byly prozkoumány teoretické koncepty, které by mohly být k tvorbě takového programu využity, a v souvislosti s výběrem jejich podmnožiny byla navržena konkrétní podoba řešení. Na základě specifik popsané aplikace byla vybrána vhodná karetní hra, blackjack. Po zprovoznění zvoleného programového prostředí byla tato hra implementována. Testováním na potenciálních uživatelích proběhlo úspěšné ověření její funkčnosti, včetně zhodnocení účelnosti použitých gest a úspěšnosti jejich rozpoznávání. Výpovědi účastníků testování svědčí o tom, že v aplikaci lze uspokojivě vykonávat všechny příslušné akce, a cíle práce tedy byly naplněny. Přesto by pochopitelně bylo možné vzniklou aplikaci dále vylepšovat a rozšiřovat. Tyto možnosti byly prozkoumány a šířeji popsány v samostatné kapitole.

Literatura

[1] Introduction: A Short History of Tabletop: Research, Technologies, and Products. MÜLLER-TOMFELDE, Christian a Morten FJELD. *Tabletops: Horizontal Interactive Displays* [online]. New York: Springer, 2010, 1-24 [cit. 2016-04-29]. Human-computer interaction series. ISBN 978-1-84996-113-4. Dostupné z: DOI:10.1007/978-1-84996-113-4_1

[2] WELCH, Gred a Eric FOXLIN. Motion tracking: no silver bullet, but a respectable arsenal. *IEEE Computer Graphics and Applications* [online]. 2002, **22**(6), 24-38 [cit. 2016-04-29]. DOI: 10.1109/MCG.2002.1046626. ISSN 0272-1716. Dostupné z: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1046626

[3] ZHANG, Zhengyou. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence* [online]. **22**(11), 1330-1334 [cit. 2016-04-27]. DOI: 10.1109/34.888718. ISSN 0162-8828. Dostupné z: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=888718

[4] DHAWAN, Amiraj a Vipul HONRAO. Implementation of Hand Detection based Techniques for Human Computer Interaction. *International Journal of Computer Applications* [online]. 2013, **72**(17), 6-13 [cit. 2016-01-26]. DOI: 10.5120/12632-9151. Dostupné z: http://arxiv.org/abs/1312.7560

[5] ELMEZAIN, Mahmoud, Ayoub AL-HAMADI, Jörg APPENRODT a Bernd MICHAELIS. A Hidden Markov Model-Based Isolated and Meaningful Hand Gesture Recognition. *International Journal of Electrical, Computer, and Systems Engineering* [online]. 2009, **3**(3), 156-163 [cit. 2016-05-02]. Dostupné z: https://www.researchgate.net/publication/228863941_A_Hidden_Markov_Model-Based_Isolated_and_Meaningful_Hand_Gesture_Recognition

[6] BLUNSOM, Phil. *Hidden Markov Models* [online]. 2004 [cit. 2016-04-29]. Dostupné z: http://digital.cs.usu.edu/~cyan/CS7960/hmm-tutorial.pdf

[7] ČERNOCKÝ, Jan. *Zpracování řečových signálů: studijní opora* [online]. Brno, 2006 [cit. 2016-05-10].
Dostupné z: http://www.fit.vutbr.cz/study/courses/ZRE/public/opora/zre_opora.pdf

[8] LEE, Hyeon-Kyu a Jin H. KIM. An HMM-Based Threshold Model Approach for Gesture Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* [online]. 1999, **21**(10), 961-973 [cit. 2016-05-10]. ISSN 0162-8828. Dostupné z: DOI:10.1109/34.799904

[9] Dynamic Time Warping. MÜLLER, Meinard. *Information Retrieval for Music and Motion* [online]. Online-Ausg. New York: Springer, 2007, s. 69-84 [cit. 2016-05-02]. ISBN 978-3-540-74048-3. Dostupné z: DOI: 10.1007/978-3-540-74048-3_4

[10] Multi-Dimensional Dynamic Time Warping for Gesture Recognition. *Thirteenth annual conference of the Advanced School for Computing and Imaging* [online]. 2007 [cit. 2016-05-02]. Dostupné z: http://mmc.tudelft.nl/sites/default/files/DTW-vASCI.pdf

Seznam příloh

Příloha 1. Implementovaná pravidla hry blackjack

Ačkoliv základní jádro hry bývá stejné, objevuje se mnoho variant hry blackjack, v nichž je např. možnost zahrát některé akce podmíněna určitou herní situací na stole nebo dokonce zcela zakázána. Existuje tedy určitá volnost ve volbě konkrétní podoby hry. Kasina mají povinnost dát hráči vysvětlení aplikovaných pravidel na požádání k nahlédnutí. Přestože se nepředpokládá žádné hazardní využití předmětu této práce, zdá se být vhodné přiložit popis implementovaných pravidel, případně upozornit na odchylky od jinak obvyklých situací. Tato množina pravidel byla zvolena zejm. s přihlédnutím ke specifikům ovládání gesty a s ohledem na co nejvyšší přehlednost uživatelského rozhraní.

Základní pravidla

Hráči hrají všichni proti krupiérovi, každý sám za sebe. Cílem hry je mít součet karet bližší číslu 21, než má krupiér, bez překročení této hranice. Hraje se se standardní sadou 52 karet. Hodnoty karet při sčítání jsou dány číselnou hodnotou, která je na nich vyobrazena, kluk, královna i král mají hodnotu 10, eso lze započítat jako 1 nebo 11.

Každému hráči jsou rozdány dvě karty, krupiér si bere jedinou. Krupiér poté postupně obsluhuje hráče jednoho po druhém ve směru hodinových ručiček. Hráč má během svého tahu následující možnosti:

- *Hit* – hráč chce další kartu. Tuto akci může hráč zahrát vždy, pokud má součet menší než 21.
- *Stand* – hráč končí svůj tah. Nedostává žádnou další kartu.
- *Double down, double* – zdvojnásobení sázku. Hráč obdrží jednu další kartu, jeho kolo pak končí. Tato karta je položena otočená o 90 stupňů na znamení toho, že hráč již nemůže žádnou další kartu dostat.
- *Split* – rozdělení hry. Tuto akci lze uplatnit pouze jako první akci po rozdání, tedy má-li hráč právě dvě karty, tyto karty navíc musí být stejné hodnoty. Karty se rozdělí, na každou z nich hráč obdrží po jedné kartě a poté hraje s těmito dvěma dvojicemi karet odděleně s každou zvlášť (jako by každou z nich dostal při rozdávání).

Pokud hráč překročí součet 21 (*bust*), okamžitě prohrává.

Po dokončení hry všech hráčů bere krupiér karty tak dlouho, dokud nemá součet 17 nebo vyšší. Poté jsou porovnány karty hráčů, kteří nepřekročili 21, a krupiéra. Pokud krupiér překročil 21, vítězí

každý ze zbylých hráčů, v opačném případě vítězí ten hráč, který má součet vyšší než krupiér, pokud má součet nižší, prohrává. Při shodě (*push*) hráč nevyhrává ani neprohrává, vsazená částka mu zůstává, výjimkou je *blackjack*, což je kombinace právě dvou karet se součtem 21 (tedy *10*, *J*, *Q* nebo *K* společně s esem), tato vítězí nad kombinací více karet, jejichž součet je 21. Při výhře s *blackjackem* je výhra vyplácena v poměru 3:2, jinak 1:1.

Porovnávání hodnot s krupiérem probíhá proti směru hodinových ručiček, nejprve jsou přijaty prohry hráčů, poté vráceny sázky u remíz a nakonec jsou vyplaceny výhry.

Další omezení a upřesnění

- *Split* není nijak omezen hodnotami karet (kromě pravidla, že hodnoty obou karet musí být stejné).
- Nelze provést vícenásobný *split.*
- Po *splitu* již nelze provést *double*.
- Po *splitu* dvou es hráč dostane jednu kartu na každé eso a jeho kolo končí. Tyto dvě obdržené karty jsou natočené v úhlu 45 stupňů ke znázornění nemožnosti dostat další karty.
- Hodnota přidané sázky při *splitu* a *doublu* je rovna původní sázce, sázka je v obou případech zdojnásobena.
- Krupiér na začátku dostane pouze jednu kartu, nehraje se s tzv. „*hole card*“.
- Krupiér zůstává stát i na „měkkém“ součtu 17, to je součet, v němž je eso započítáno s hodnotou 11.
- Hráči není umožněno vzdát hru (*surrender*).
- Nelze vsadit sázku na to, že krupiér dostane *blackjack* (tzv. pojištění, *insurance*).
- Nelze uzavírat žádné sázky na kombinace (*side bets*).
- Není dovoleno provádět oddělené vklady sázek více hráčů na jedné pozici (*back betting*).
- Hraje se s jedním balíčkem karet. Okamžitě po jeho vyčerpání je připraven balíček nový.