**Mendelova univerzita v Brně**
**Provozně ekonomická fakulta**

# Šachový automat s využitím manipulátoru Katana

## Diplomová práce

**Vedoucí práce:**

**Prof. RNDr. Ing. Jiří Šťastný, Csc.** **Bc. Marcel Vytečka**

**Brno 2013**

Rád bych poděkoval vedoucímu této diplomové práce, panu prof. RNDr. Ing. Jiřímu Šťastnému, CSc., za jeho hodnotné rady a čas, který mi věnoval. Dále bych chtěl poděkovat Ing. Janu Kolomazníkovi, Ing. Vítu Ondrouškovi, Ph.D., Ing. Jaromíru Landovi, Ing. Tomášovi Koubkovi a všem kolegům za připomínky a rady, související s problematikou této práce.

Prohlašuji, že jsem tuto diplomovou práci vyřešil samostatně s použitím literatury, kterou uvádím v seznamu.

V Brně dne 30. dubna 2013 _____

**Abstract**

Vytečka, M. *Katana manipulator used for Chess automat*. Diploma thesis Brno: 2013.

This diploma thesis describes design and implementation of a chess automat. Results of this work are modules for image processing, control of manipulator, visualization of a game scene, chess algorithm and control of an electromagnet. Process of design and implementation of this system is detailed described in this work. Keywords
OpenCV, Chess, Katana, Irrlicht

**Abstrakt**

Vytečka, M. *Šachový automat s využitím manipulátoru Katana*. Diplomová práce. Brno: 2013.

Práce se zabývá návrhem a realizací šachového automatu. Výsledkem práce je systém, v němž je integrován modul pro zpracování obrazu, řízení manipulátoru, vizualizaci herní scény, šachového algoritmu a ovládání elektromagnetu. Proces návrhu a implementace systému je v práci detailně popsán.

**Klíčová slova**

OpenCV, šachy, Katana, Irrlicht.

# Obsah

**1 Úvod a cíl práce 10**

1.1 Úvod ... 10

1.2 Cíl práce ... 11

**2 Požadavky na řešení 12**

2.1 Základní specifikace požadavků ... 12

2.2 Požadavky na vzorový projekt pro výuku ... 13

**3 Analýza současného stavu 14**

3.1 Specifikace omezení vyhledávání ... 14

3.2 Přehled nalezených řešení ... 14

3.2.1 Marine Blue: A low-Cost chess Robot ... 14

3.2.1 Chess robot system: A multi-disciplinary experiencce in automation 17

3.2.2 Gambit: A robust chess-playing robotic system ... 19

3.1 Zhodnocení nalezených řešení ... 21

**4 Manipulátor 22**

4.1 Požadavky na manipulátor v šachovém automatu ... 22

4.1.1 Manipulátor Katana ... 22

Základní charakteristika ... 22

Možnosti programování ... 22

Technický popis ... 23

Souřadný systém ... 24

Použití v úloze hraní šachů ... 25

4.1.2 Možné alternativy ... 25

Manipulátor MELFA RV2-AJ ... 25

Možnosti programování ... 25

Použití v úloze hraní šachů ... 26

**5 Zpracování obrazu 27**
5.1 Hardware ..... 27
5.2 Software ..... 28
5.2.1 OpenCV ..... 28
Rozhraní ..... 28
Použití v úloze hraní šachů ..... 28
5.2.2 Možné alternativy ..... 28
VisionLab ..... 28

**6 Šachové algoritmy 29**
6.1 Huo Chess ..... 29
6.2 StockFish ..... 29
6.3 GNU Chess ..... 29

**7 Hardwarové vybavení 30**
7.1 Vývojová platforma Arduino ..... 30
7.2 Prototypová deska a spínací relé ..... 30

**8 Vizualizace herní scény 31**
8.1 3D vizualizace ..... 31
8.1.1 Irrlicht 3D engine ..... 31
8.1.2 OGRE 3D engine ..... 31

**9 Realizace šachového automatu 32**
9.1 Fyzické upevnění kamery na manipulátor Katana ..... 32
9.2 Popis principu celého řešení ..... 33
9.3 Propojení SW pro snímání obrazu se systémem pohybu a systémem vizualizace ..... 33
9.3.1 Popis modulů programu ..... 33
9.3.2 Běh programu ..... 35
9.4 Kalibrace kamery ..... 36
9.5 Vyhledání vzorů v obraze ..... 38
9.6 Převod souřadných systémů ..... 39
9.7 Snímání herní šachovnice ..... 41
9.8 Počáteční inicializace figur ..... 41

9.8.1 Rozlišení černých a bílých figur ..... 42
9.9 Zjištění pozic figur po tahu živého hráče ..... 42
9.9.1 Kontrola počtu nalezených figur ..... 42
9.9.2 Určení pozice figur ..... 43
9.9.3 Kontrola počtu změn v poloze figur ..... 43
9.9.4 Neplatný tah ..... 44
9.10 Vytvoření vzorového projektu pro výuku ..... 44
9.11 Modelování a tisk šachovnice a figur na 3D tiskárně ..... 44
9.12 Fyzická úprava šachovnice a figur ..... 45
9.12.1 Vytvoření šachovnice ..... 45
9.12.2 Úprava figur ..... 46
9.13 Fyzická úprava uchopovače manipulátoru ..... 47
9.14 Ovládání elektromagnetu ..... 48
9.15 Tah manipulátoru ..... 49
9.16 Vizualizace herní scény ..... 49

**10 Ověření spolehlivosti řešení 51**
10.1 Spolehlivost zpracování obrazu ..... 51
10.2 Spolehlivost manipulace ..... 51

**11 Diskuze 52**

**12 Závěr 54**

**13 Literatura 55**

**A Šachový automat při prezentaci 58**

**B Arduino zdrojový kód 59**

**C Ukázka podobných řešení 60**

**D Prototyp figur pro vývoj 61**

# Seznam obrázků

Obr. 1 Základní schéma požadavků na šachový automat ..... 12
Obr. 2 Klasifikace pixelů při použití HSB barevného prostoru. ..... 15
Obr. 3 Horizontální přesun čtvrtého segmentu robotu při pohybu. ..... 16
Obr. 4 Architektura systému Chess robot systém. ..... 17
Obr. 5 Ukázka systému vidění ..... 18
Obr. 6 Schéma stupňů volnosti manipulátoru. ..... 19
Obr. 7 Vyhledání figury v obraze a rozlišení jejího typu. ..... 20
Obr. 8 Rozměry manipulátoru ..... 23
Obr. 9 Univerzální uchopovač ..... 24
Obr. 10 Souřadné systémy manipulátoru. ..... 24
Obr. 11 Manipulační ruka Melfa RV-2AJ ..... 25
Obr. 12 Webová kamera Microsof LifeCam studio ..... 27
Obr. 13 Vývojová platforma Arduino ..... 30
Obr. 14 Upevnění kamery na manipulátor. ..... 32
Obr. 15 Schématické znázornění celého řešení šachového automatu ..... 33
Obr. 16 Schématické znázornění jednotlivých modulů šachového automatu ..... 34
Obr. 17 Diagram aktivit programu šachového automatu ..... 36
Obr. 18 Kalibrace kamery pomocí šachovnice. ..... 38
Obr. 19 Nalezení vzorů při inicializaci hry. ..... 39
Obr. 20 Kružnice pro převod souřadných systémů. ..... 39
Obr. 21 Zobrazení nalezených kružnic a jejich středů. ..... 40
Obr. 22 Nalezení rohů šachovnice ..... 41
Obr. 23 3D model figury věž. ..... 45
Obr. 24 3D model jedné komponenty šachovnice. ..... 45
Obr. 25 Umístění magnetu na figuru. ..... 46
Obr. 26 Upravené figury zasazené do šachovnice. ..... 46
Obr. 27 Upevnění elektromagnetu na kulovém adaptéru. ..... 47
Obr. 28 Schéma zapojení obvodu pro spínání elektromagnetu. ..... 48
Obr. 29 3D scéna šachovnice ..... 50
Obr. 30 Šachový automat při prezentaci Mendelovy univerzity v Brně ..... 58
Obr. 31 Šachový automat při prezentaci Mendelovy univerzity v Brně ..... 58
Obr. 32 Šachový robotický multidisciplinární systém ..... 60
Obr. 33 Šachový robot gambit při hře ..... 60
Obr. 34 Prototyp figur pro vývoj software před 3D tiskem. ..... 61

# Seznam tabulek

Tabulka 1 Technické parametry manipulátoru ..... **23**
Tabulka 2 příklad rozložení figur ve vnitřní reprezentaci ..... **43**
Tabulka 3 Procentuální úspěšnost při zpracování obrazu ..... **51**
Tabulka 4 Procentuální úspěšnost při manipulaci s figurami ..... **51**

# 1 Úvod a cíl práce

## 1.1 Úvod

Automatizace procesů je důležitou součástí průmyslové výroby. Pro robotická pracoviště jsou charakteristické takové vlastnosti jako vysoká rychlost, vysoká přesnost a provoz téměř bez přestávky. Tyto vlastnosti vedou ke zvýšení spolehlivosti a snížení nákladů na provoz pracoviště. Největší nevýhodou je poměrně vysoká pořizovací cena. Také při změně výrobního procesu a údržbě robotických zařízení je vyžadován zásah odborníka. I přes tyto nevýhody si však nelze představit moderní průmyslový podnik bez automatizovaného pracoviště.

Manipulátory s nižším výkonem a vysokým zabezpečením mohou sdílet pracovní prostor s člověkem. Naopak manipulátory s vysokým výkonem, nebo s jiným omezením, musejí být v chráněném prostoru. Tímto prostorem může být vyhrazený pracovní prostor bez přítomnosti člověka, ochranná klec nebo uzavřená místnost. Stacionární roboty Katana 300s spadají do první kategorie, a proto mohou pracovat v blízkosti člověka. Jsou konstruovány pro lehčí manipulační úlohy a jsou také vhodné do školního prostředí pro výuku principů automatizace.

Mnoho průmyslových manipulátorů je naprogramováno staticky. Mají naučené pohyby, které se opakují. Nemají žádné senzory a nezaznamenají změnu v okolním prostředí. Spoléhají se na přesné naplánování procesu. Při jakékoli nežádoucí změně v prostředí nejsou schopny adekvátně zareagovat a zpravidla nedokončí požadovanou činnost. V horším případě se mohou dokonce poškodit. Oproti tomu roboty vybavené senzory pro snímání prostředí a manipulovaných prvků, mohou být řízeny adaptivně. Senzory mohou být různého typu. Mohou měřit teplotu, tlak, přítomnost objektu, materiál objektu a mnoho dalších vlastností okolního prostředí. Pro snímání objektů lze také využít kameru. Vyhodnocením snímků z kamery je možné, mimo jiné, získat informace o počtu objektů, vzdálenosti, rozměrech a natočení. Největší výhodou kamerového snímání je možnost použití robotů i v procesech, kde nelze s absolutní spolehlivostí předvídat polohu objektů, se kterými má robot manipulovat. Příkladem mohou být roboty, které na konci výrobního procesu balí potravinářské výrobky do obalů a připravují na výdej. Zpravidla jsou tyto výrobky náhodně rozmístěny na pohyblivém pásu výrobní linky a robot si musí poradit s jejich transportem do připravených přepravek. Může nahradit pracovníka stojícího u pohyblivého pásu a vykonávajícího monotónní práci. Tuto práci vykoná rychleji a hygieničtěji.

Manipulátor Katana, použitý v této práci, je vybaven kamerou pro snímání herní scény a pohybuje se v závislosti na změnách prostředí. Souřadnice pro pohyb manipulátoru na pozici šachových figur jsou dynamicky vypočítané v závislosti na informacích přicházejících ze snímků herní scény.

První zmínka o šachovém automatu pochází z 18. Století. Byl jím stroj označovaný jako Turk. Zkonstruoval jej německý vynálezce Wolfgang von Kempelen

v roce 1770. Skládal se z velkého stolu, za kterým se nacházelo vrchní torzo muže v kostýmu Turka. Stůl měl mnoho dveří, po jejichž otevření se odhalilo složité soukolí ozubených kol a mechanických dílů. Stroj byl prezentován jako automaton, předchůdce dnešních robotů. Přestože se nakonec ukázalo, že ve stole byl prostor pro člověka, který celý stroj ovládal, stal se Turk fenoménem v oblasti šachových automatů. Originál byl zničen při požáru v roce 1854 (WiseGEEK, 2003).

S rozvojem robotiky ve 20. století se objevují první, počítačem řízené, šachové automaty. Automatizované šachové úlohy lze rozdělit na tři hlavní části: systém detekce tahu protivníka, šachový algoritmus a manipulační systém. Detekce tahu je v mnohých případech řešena přímo hrací deskou. Již menší počet řešitelů se uchýlil ke zpracování obrazu šachovnice.

Dnes je mnoho šachových algoritmů volně stažitelných a zdarma. Nejčastěji jde o konzolové aplikace, připravené pro implementaci do grafického uživatelského rozhraní.

Systém manipulace s figurami je nejspecifičtější částí těchto úloh. Existuje mnoho druhů manipulátorů a pro každý je systém manipulace unikátní. Dále v práci je popsán způsob řešení systému manipulace pro robot Katana 300s.

## 1.2 Cíl práce

Cílem této diplomové práce je návrh a implementace systému pro fyzické hraní hry šachy s využitím programovatelného manipulátoru Katana. Součástí tohoto cíle je implementace šachového algoritmu a systému pro zpracování obrazu do systému řízení. Dále návrh šachových figur a vytvoření 3D vizualizace šachovnice k použité aplikaci.

Dílčím cílem je využití části řešení pro vytvoření projektu pro pohyb manipulátoru Katana na základě zpracování obrazu z kamery. Tento projekt poslouží jako základ k vytvoření úloh do cvičení předmětů vyučovaných na Provozně ekonomické fakultě Mendelovy univerzity v Brně.

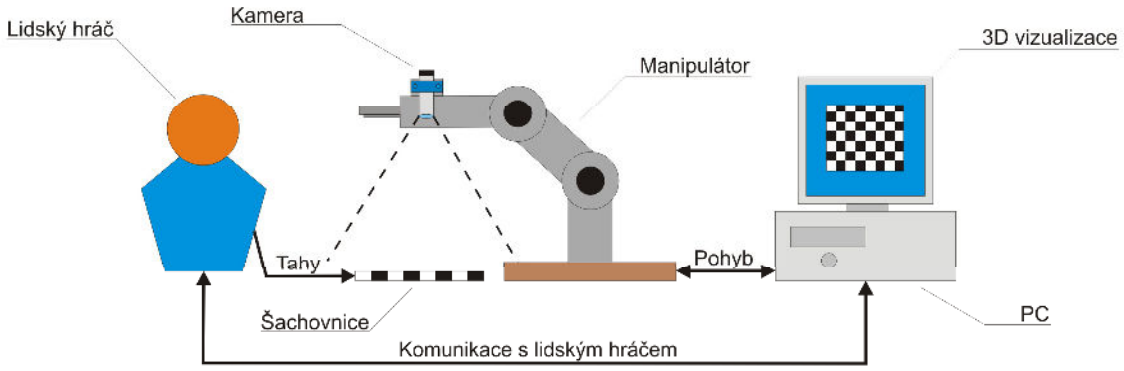
# 2 Požadavky na řešení

## 2.1 Základní specifikace požadavků

Šachový automat bude sloužit především jako prostředek pro prezentaci univerzity a jejích oborů zaměřených na automatizaci a řízení. Základním požadavkem je hra šachů proti lidskému hráči. Každá akce systému bude demonstrativně zobrazena na monitoru počítače, tabletu, případně projektoru. Zejména jde o snímaný obraz z kamery, detekce figur, zobrazení souřadnic pohybu manipulátoru a 3D reprezentace herní scény korespondující s aktuálním rozložením figur na šachovnici.

O pohyb figur na šachovnici při tahu počítačem řízeného hráče se bude starat manipulátor řízený počítačem. Kamera namontovaná na rameni manipulátoru bude zajišťovat lepší transportovatelnost celého automatu. Celý systém bude možné po transportu uživatelsky kalibrovat bez zásahu do zdrojového kódu programu. Tuto kalibraci bude provádět operátor. Po této kalibraci bude program již ovládat lidský hráč.

Program bude v průběhu hry vyzívat hráče ke dvěma fyzickým aktivitám. První z nich je výzva k provedení tahu. Druhou je výzva k opravě neplatného tahu. Bude také žádat dvě vstupní informace. První z nich je informace o dokončení tahu lidského hráče. Bude reprezentována stiskem požadované klávesy. Druhá informace se týká výběru role při promoci pěšce. Pokud pěšec dojde na opačný konec šachovnice, bude mít lidský hráč na výběr ze čtyř různých rolí. Každá role bude reprezentována číslem. Po zadání a potvrzení volby se pěšec přizpůsobí nové roli jak v programu, tak ve 3D vizualizaci. Podle pravidel hry šachy může hráč hrát až s devíti dámami na šachovnici.




Obr. 1 Základní schéma požadavků na šachový automat

## 2.2 Požadavky na vzorový projekt pro výuku

Vzorový projekt pro výuku bude integrovat systém zpracování obrazu se systémem pohybu manipulátoru v prostředí Visual Studio 2010. Základní funkcionalitou bude vyhledání vzorů v obraze a pohyb manipulátoru na tyto souřadnice. Musí být umožněno rozšíření úlohy o manipulaci s prvky. Nejdůležitějším požadavkem je správné přednastavení bezpečnostních zásad pro práci s manipulátorem. Mezi tyto zásady a nastavení patří zejména rychlost pohybu a zablokování při nárazu.

Vzorový projekt bude využívat upevněné kamery na manipulátoru stejně jako úloha hraní šachů. Implementované postupy pro zpracování obrazu a přepočet souřadnic budou také využity v obou projektech.

# 3 Analýza současného stavu

## 3.1 Specifikace omezení vyhledávání

Při vyhledávání existujících řešení problematiky šachových automatů nebyl brán zřetel na úlohy, získávající informaci o přesunu šachových figur z podstavy šachovnice. Tato řešení jsou zaměřena výhradně na návrh šachového algoritmu a programování manipulátoru. Nedochází u nich k žádnému zpracování obrazu, získaného z kamery. V současné době nejznámější robot využívající senzorickou hrací desku je robot ChessKA, kterého sestavil Konstantin Kosteniuk. Stal se světovým robotickým šachovým mistrem v roce 2012 (Chess-king, 2013).

Dále byla z vyhledávání vyřazena řešení, která nepoužívají pro řešení manipulátor s minimálně čtyřmi stupni volnosti. Do této kategorie spadají manipulátory typu stacionární kontejnerový překladač a všechny typy manipulace pod hrací deskou, například magnetické. Příkladem tohoto typu robotu může být robochess. Jeho konstruktérem je Ebrahim Jahandar (Jahandar, 2013).
Analýza je zaměřena na projekty, řešící tři základní složky této problematiky integrované v jednom systému. Zpracování obrazu z kamery pro získání rozložení figur na šachovnici. Implementace šachového algoritmu. A řešení pohybu manipulátoru při tahu počítačového hráče.

## 3.2 Přehled nalezených řešení

### 3.2.1 Marine Blue: A low-Cost chess Robot

MarineBlue je plně autonomní robot, zkonstruovaný pro účely výuky. Jeho konstruktéři jsou David Urting a Yolande Berbers. Prvotním impulsem pro vytvoření MarineBlue šahového automatu bylo pro autory zjištění, že existující automaty jsou velké a drahé, nebo jsou limitováni omezenou pohyblivostí a funkcemi. Jejich řešení se je složeno z několika komponent, které jsou níže popsány (Urting, 2013).

#### Šachová hrací deska a figury

Standardní šachová hrací deska má rozměry 40 až 50 milimetrů. Deska použitá u tohoto automatu byla zmenšena na 30 milimetrů. Jako hrací figury slouží standardní sada dřevěných šachů. U šachovnice i u figur byla změněna barva jednotlivých součástí z důvodů snímání kamerou. Byly použity barvy, které se navzájem vylučují (Urting, 2013).

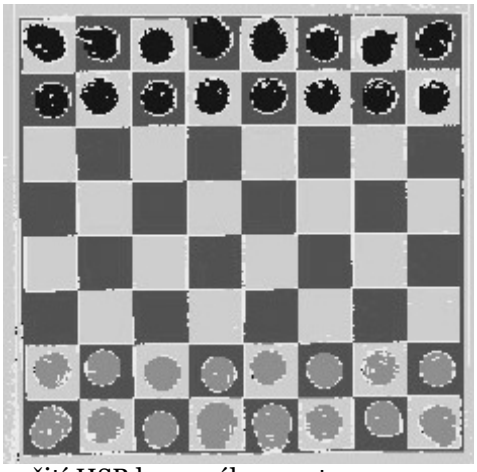
## Kamera

Pro snímání herní scény byla použita vysoce kvalitní kamera Sony DFW-VL500, umístěná jeden metr nad hrací deskou. Díky tomuto umístění je minimalizováno perspektivní zkreslení. Kvalitní snímky jsou nezbytné pro analýzu obrazu (Urting, 2013).

## Zpracování obrazu

Algoritmus pro zpracování obrazu je implementován ve třech vrstvách. Vrstva klasifikace pixelů, vrstva hrací desky a vrstva šachovnice.

Vrstva klasifikace pixelů obdrží RGB (Red, Green, Blue) bitmapový snímek z kamery. Při kalibraci jsou pixely rozlišeny do čtyř tříd. Světlý čtverec, tmavý čtverec, světlá figura a tmavá figura. Nejlepších výsledků je dosaženo pomocí přechodu na HSB (Hue, Saturation, Brightness) barevný prostor. Nalezené výsledky předá vyšší vrstvě (Urting, 2013).



Obr. 2 Klasifikace pixelů při použití HSB barevného prostoru.
Zdroj: Převzato od Urtinga (2013).

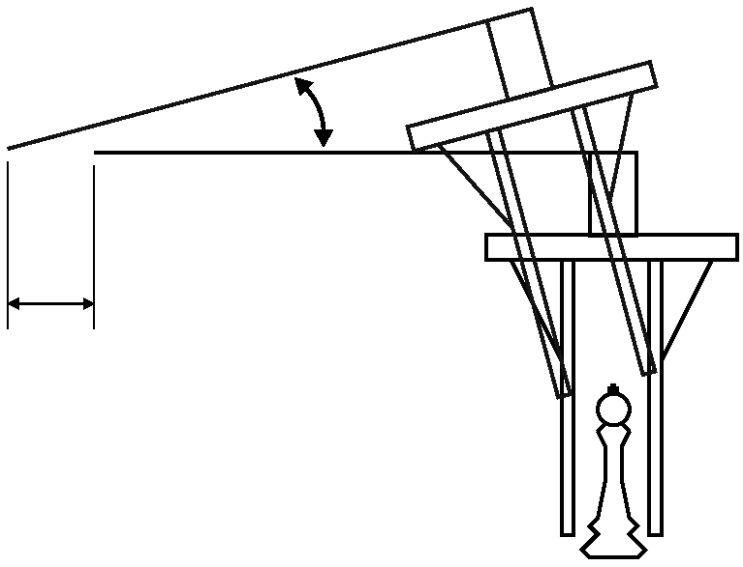
Vrstva hrací desky obdrží od předchozí vrstvy matici s klasifikací pixelů. Určí pozici šachovnice a také obsazenost jednotlivých polí. Výstupem této vrstvy je seznam všech polí s informací o obsazenosti figurou určité barvy.
Vrstva šachovnice udržuje informaci o předchozím rozložení figur na šachovnici. Převezme výstup z předchozí vrstvy, obsahující informaci o aktuálním rozložení. Díky informaci o předchozím tahu nemusí zjišťovat druh figury přímo z obrazu, ale jednoduše jej dopočítá z rozdílu minulého a aktuálního tahu. Vrstva šachovnice je schopná detekovat i tah rošády. Pokud však hráč zamění dvě figury stejné barvy, není tento podvod možné detekovat, protože zpracování obrazu je založené na určení pozice figury jen podle barvy (Urting, 2013).

## Robotická ruka

Pro pohyb figur je použit robot Robix RC-6. Jedná se o robota pro výukové účely poháněného soustavou servomotorů. Robot je připojen přes paralelní port a příkazy mu jsou posílány přes Robix-dependent skriptovací jazyk (Urting, 2013).

## Ovládání robotu

Komponenta software ovládání robotu se stará o převod vysokoúrovňových příkazů (pohyb figury ze souřadnice C-1 na C-2) na nízko úrovňové (nastavení servomotorů robotu).
Robot ví, kde se nachází šachovnice v jeho souřadném systému. Proto nemusí probíhat kalibrace mezi kamerou a robotem. Po zadání požadované změny je pohyb robotu dopočítán inverzní kinematikou. Pro daný koncový bod jsou zjištěny všechny existující řešení nastavení servomotoru a je vybrána nejvhodnější. Jako nejvhodnější je zpravidla vybrána možnost s nejmenším nutným pohybem všech servomotorů (Urting, 2013).



Obr. 3 Horizontální přesun čtvrtého segmentu robotu při pohybu.
Zdroj: převzato od Urtinga (2013).

### Šachový počítač

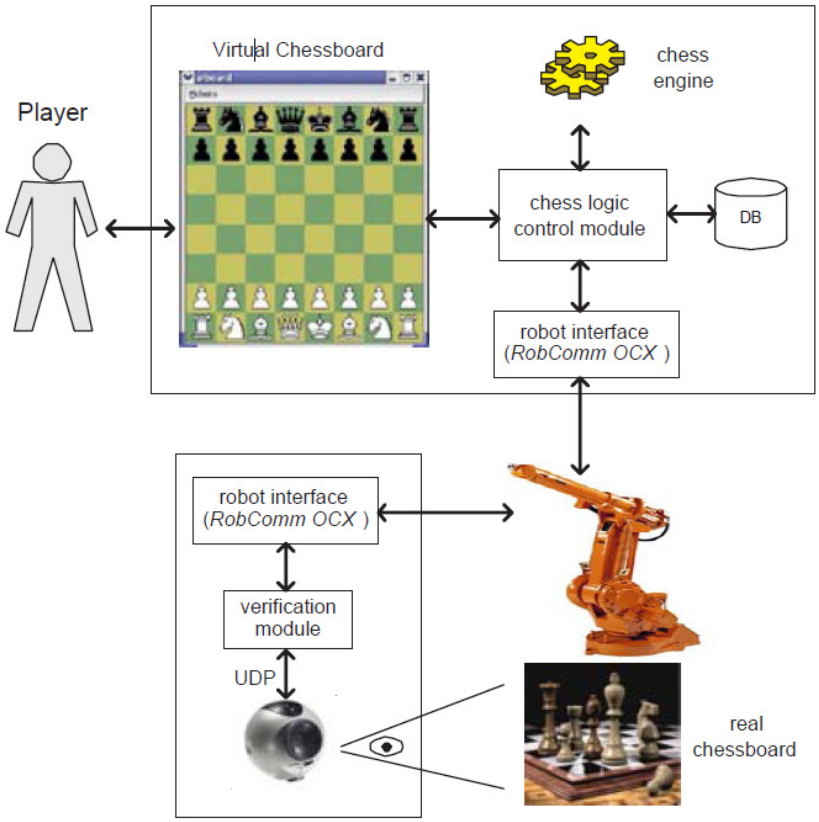
Celé řešení je implementováno v prostředí operačního systému Windows v programovacím jazyce C++ za použití MS Visual studio. Jako šachový algoritmus je použit GNU Chess. Protože je řešení postaveno na třech nezávislých modulech, je poměrně jednoduché nahradit tyto součásti za jiné. Zejména je tato vlastnost užitečná při změně hardwarového systému. Více o tomto řešení lze nalézt v Urting (2013).

## 3.2.1 Chess robot system: A multi-disciplinary experiencce in automation

Šachového robota, jehož konstruktéři jsou José Goncalves, José Lima a Paulo Leitao, lze použít pro vzdálené hraní šachů. Uživatel ovládá program přes grafické uživatelské rozhraní. Robot reaguje na tyto povely a pohybuje figurami (Goncalves, 2013).

### Architektura systému

Systém je složen z pěti modulů. Šachovnice, robot, softwarová aplikace a umělé vidění.




Obr. 4 Architektura systému Chess robot systém.
Zdroj: převzato od Goncalvese (2013).

## Softwarová aplikace

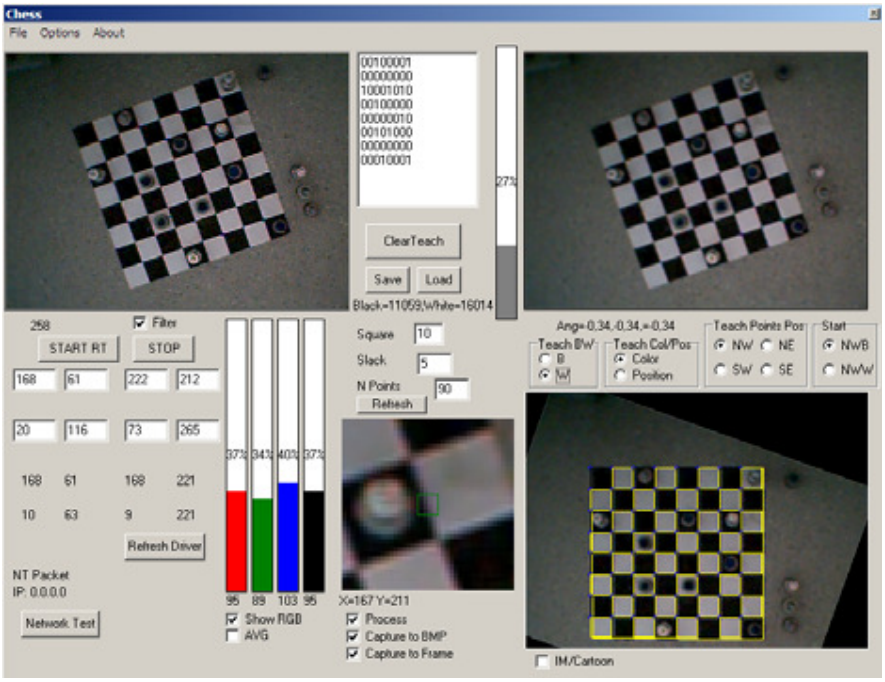
Softwarová aplikace obsahuje rozhraní pro lidského hráče, ovládání robotu a šachový algoritmus. Grafické uživatelské rozhraní dovoluje lidskému hráči posouvat figurami na virtuální šachovnici systémem „chyť a pusť“. Každý tento tah je vyhodnocen šachovým algoritmem. Pokud je vyhodnocen jako platný, jsou vypočítány parametry pro pohyb a předány robotu (Goncalves, 2013).

## Robot

Technologie použitá pro pohyb figur je ABB IRB 1400 robot, vybavený pneumaticko-elektrickým uchopovačem. Pokud robot dostane od softwarové aplikace příkaz k pohybu, kontaktuje modul umělého vidění, který mu zprostředkuje informaci, zda na cílovém poli není figura a může tedy bezpečně táhnout (Goncalves, 2013).

## Umělé vidění

Modul umělého vidění obsahuje verifikační mechanizmus, komunikující přímo s rozhraním robotu. Analýza obrazu v tomto projektu probíhá z důvodu ověření, zda cílové pole není obsazeno. S fyzickými figurami pohybuje vždy robot na příkaz lidského hráče, nebo šachového algoritmu. Není tedy nutné rozpoznávat aktuální rozložení na hrací desce.




Obr. 5 Ukázka systému vidění
Zdroj: Převzato od Goncalvese (2013).

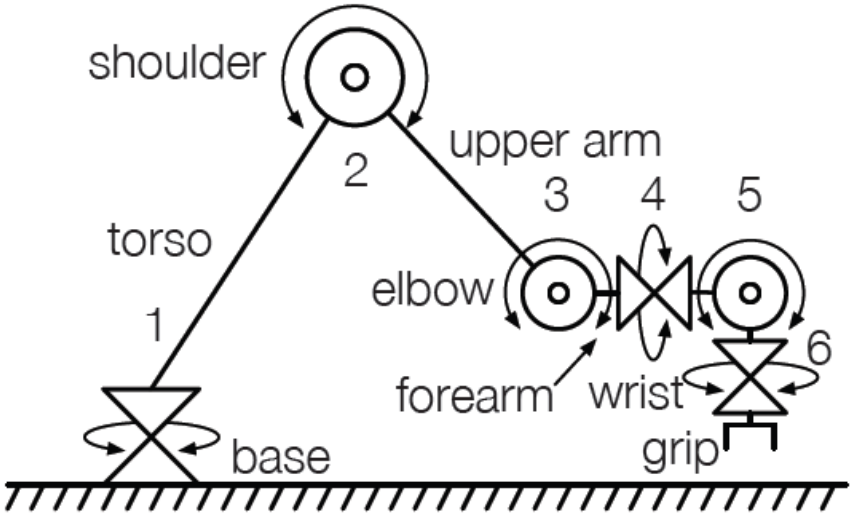
Zpracování snímku z kamery začíná odstraněním šumu pomocí Gaussova filtru. Pokračuje odstraněním nepřesností získaných při přenosu snímku. Hlavní aktivitou umělého vidění je rozpoznání šachovnice, detekce jejího natočení a nalezení obsazených polí. Protože šachovnice má známé rozměry, je možné tyto informace dopočítat. Rozlišení jednotlivých figur je řešeno pomocí barevné analýzy. Proto je nutné kameru vždy před použitím kalibrovat. I menší změna nasvětlení může analýzu znehodnotit. Po nalezení informace o obsazenosti cílového pole je robotu dovoleno provést tah. Více o tomto řešení lze nalézt v Goncalves (2013).

### 3.2.2 Gambit: A robust chess-playing robotic system

Robotický systém Gambit je z nalezených řešení nejrobustnější. Pro pohybový modul byl navržen a zkonstruován nový typ robotu s šesti stupni volnosti včetně řídicího systému. Pro detekci tahů na herní desce je použita kamera. Další kamera je umístěna přímo v uchopovači. Systém dokáže hrát s mnoha druhy šachových figur. S uživatelem systém komunikuje pomocí přirozeného jazyka. Lidský hráč je vždy srozuměn s potvrzením svého tahu a je mu oznámen plánovaný tah robotu. Pokud se stane, že robot upustí figuru nebo se mu jí nepodaří správně umístit na správné místo, promluví na hráče a vyžádá si jeho pomoc při nápravě této situace (Matuszek, 2013).

#### Manipulátor

V systému je použita manipulační ruka navržená autory. Nejedná se o sériově vyráběný model. Disponuje šesti stupni volnosti a paralelním uchopovačem. Základní pohyb ramene na pozici zajišťuje trojice harmonických motorů s přesností na tisíciny stupně. Na obrázku jsou označeny čísly 1, 2 a 3. Zápěstí manipulátoru je tvořeno třemi motory s přesností na desetiny stupně. Tyto motory jsou označeny čísly 4, 5 a 6.




Obr. 6 Schéma stupňů volnosti manipulátoru.
Zdroj: Převzato od Matuszeka, (2013).

Koncový uchopovač lze lehce zaměnit za jiný koncový prvek, odvíjející se od charakteru úlohy. Pořizovací cena součástí manipulátoru je osmnáct tisíc dolarů (Matuszek, 2013).

### Kamerový systém

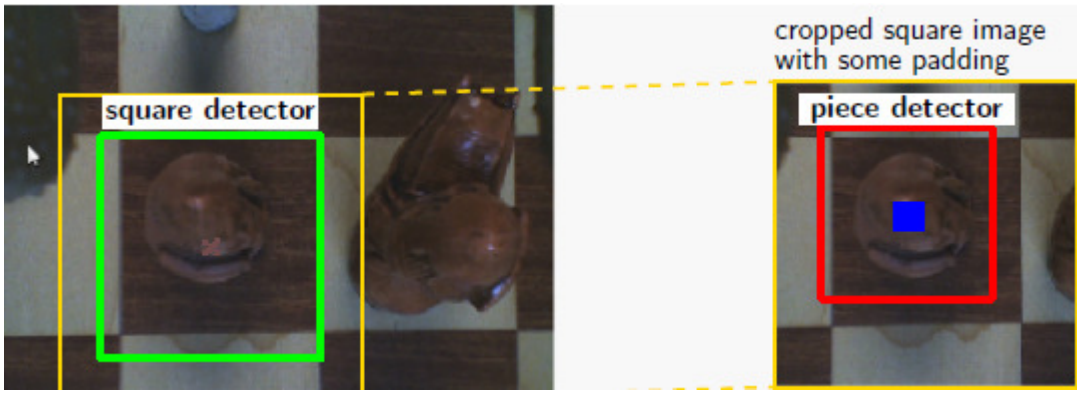
Hlavní kamera, technologicky identická s Xbox Kinect, je umístěna na horní části ramene manipulátoru. Menší kamera je integrovaná do uchopovače manipulátoru. Protože cílem autorů bylo vyvinout samostatný integrovaný systém, zvolili umístění kamery přímo na manipulátoru. Integrace menší kamery byla možná jen s použitím stíněného kabelu a USB zesilovače za prvními dvěma motory.

Prvním krokem při zpracování obrazu z hlavní kamery je detekce šachovnice. Algoritmus detekuje rohové body šachovnice. V závislosti na pozici bodů nalezne polohu šachovnice vzhledem ke kameře. Kalibrace probíhá kontinuálně a lze s hrací deskou pohybovat i během hry. Algoritmus je schopný detekovat šachovnici, i pokud jsou některé body zastíněny rukou lidského hráče nebo figurami. Dále následuje rozlišení polí na prázdné a obsazené figurou. Po určení rozložení je toto zasláno šachovému algoritmu a ten vyhodnotí správnost tahu. Pokud je tah nesprávný, lidský hráč je vyzván k nápravě. Jako šachový algoritmus je v tomto řešení použit GNU Chess (Matuszek, 2013).

### Rozpoznání typu figury

V době psaní této práce potřebuje Gambit na začátku vědět rozložení figur. Systém pro rozlišení typu figury je již navržen a popsán autory tohoto řešení.

Pro detekci typu figury bude sloužit menší kamera, umístěna v uchopovači manipulátoru. Kamera si vytvoří trénovací množinu snímků v různém natočení kamery. Algoritmus si vytvoří z těchto snímků vzory, které poté použije pro testování typu figury (Matuszek, 2013).




Obr. 7 Vyhledání figury v obraze a rozlišení jejího typu.
Zdroj: Převzato od Matuszeka (2013).

## 3.1 Zhodnocení nalezených řešení

Z nalezených řešení, obsahujících modul manipulátoru, zpracování obrazu a šachového algoritmu, se jako nejsofistikovanější jeví Gambit, uvedený v kapitole 3.2.2. Tomu odpovídají i náklady na celý projekt. Zejména kamera pro zpracování obrazu je na vysoké úrovni. Bohužel prostředky, vyčleněné pro tento projekt, neumožňují toto vybavení pořídit. Rozpoznávání klasických šachových figur proto není předmětem této práce. Tato možnost rozšíření je rozvinuta v diskuzi.

Také návrh vlastního manipulátoru umožnil větší pohyblivost po pracovní ploše, než je možné dosáhnout s manipulátory, které jsou pro tento projekt k dispozici.

Moduly pro řízení manipulátoru u všech nalezených řešení jsou velice specifické z důvodu použití různého hardware.

# 4 Manipulátor

## 4.1 Požadavky na manipulátor v šachovém automatu

Protože při provozu automatu bude člověk v těsné blízkosti manipulátoru, nejdůležitějším požadavkem je bezpečnost. Manipulátor musí být vybaven funkcí blokování motorů při kolizi. Dále musí být možné nastavit rychlost manipulátoru, zrychlení a bezpečnou hranici pro kolizi.

### 4.1.1 Manipulátor Katana

#### Základní charakteristika

Roboty Švýcarské společnosti Neuronics AG Katana verze KatHD300s jsou bezpečné manipulátory, které mohou sdílet pracoviště s člověkem. Pro provoz těchto robotů v bezprostřední blízkosti člověka není zapotřebí ochranné klece. V této práci je použit konkrétní model Katana 6M180. Tento model lze připojit k řídicímu počítači pouze prostřednictvím sériové linky RS-232. Protože je tento robot konstruován pro práci v blízkosti člověka, je vhodný na úlohu hraní šachů (Neuronics, 2011).

#### Možnosti programování

Katana Native Interface (dále jen KNI)

KNI je soubor knihoven, který slouží k ovládání robotu Katana v programovacím jazyce C++, popřípadě Pythonu. KNI pracuje ve třech vrstvách. Vrstva komunikačního rozhraní CDL, Vrstva komunikačního protokolu CPL a vrstva Katana Robot Model KML. Více detailních informací o softwarové knihovně lze nalézt v Neuronics (2006)

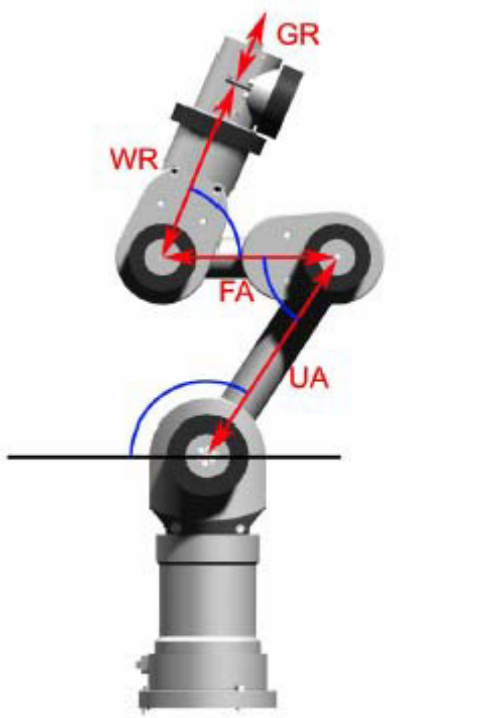
Katana4D

Standardně dodávaný software společnosti Neuronics AG. Pomocí grafického uživatelského rozhraní lze jednoduše nastavit základní funkce robotu. Je možné také číst hodnoty senzorů a vytvářet komplexní programy pro řízení. Podporuje role administrátor a uživatel. Tento software umožňuje správu a ovládání robotu i lidem, kteří neumějí pracovat s programovacím jazykem C++.

Control Web

Pro řízení robotu Katana je možné také využít prostředí Control Web. Pracuje s technologií ActiveX, která zajišťuje propojovací rozhraní mezi manipulátorem a prostředím Control Web. Více o tomto systému řízení můžete najít v Rouš (2009).

## Technický popis

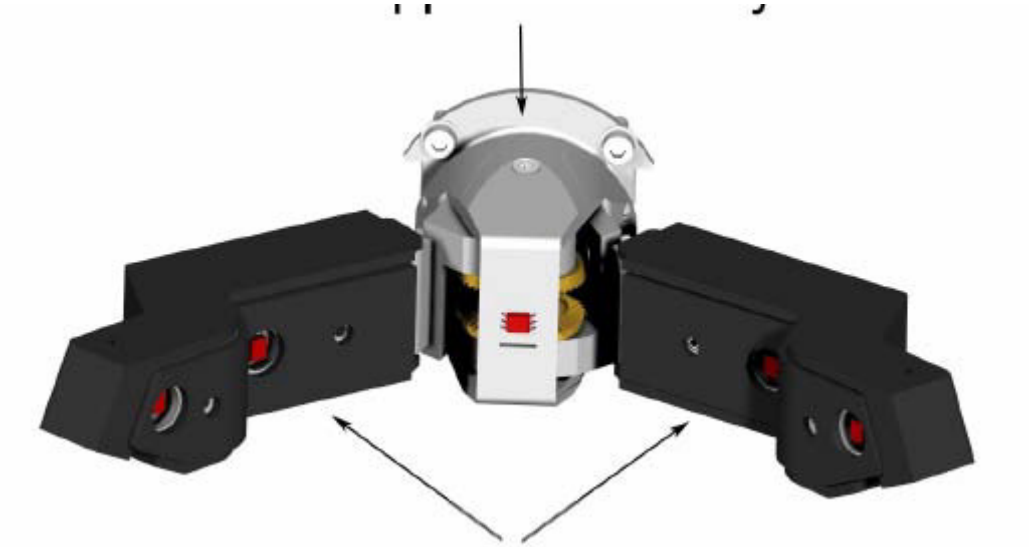



Obr. 8 Rozměry manipulátoru
Zdroj: Převzato od Neuronics (2006)

Tabulka 1 Technické parametry manipulátoru

| | |
|---|---|
| DOF | 5 |
| Max. výška [mm] | 854 |
| Úhly vyosení [°] | M1: 0 |
| | M2: 124.25 |
| | M3: 52.7 |
| | M4: 63.5 |
| | M5: 8.5 |
| Délka ramene [mm] | UA: 190 |
| | FA: 139 |
| | WR: 185 |
| | GR: 130 (s uchopovačem) |
| Operační rozsah [°] | M1: 345.7 |
| | M2: 140 |
| | M3: 241.5 |
| | M4: 232 |
| | M5: 332.2 |
| | M6: 140 (čelisti) |

Zdroj: Převzato od Neuronics (2006).

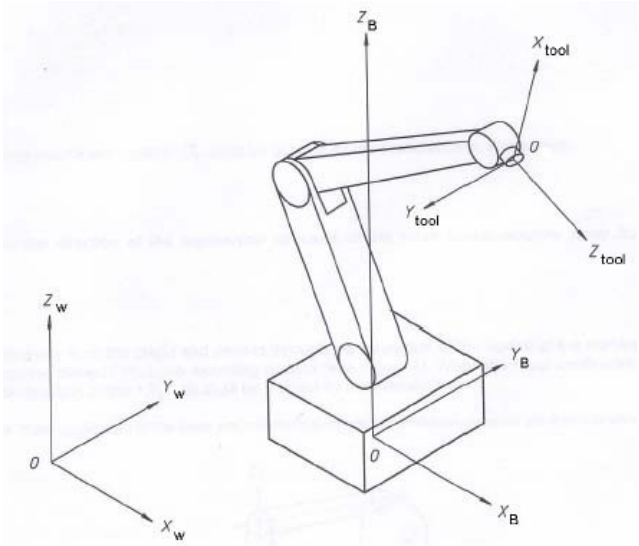


Obr. 9 Univerzální uchopovač
Zdroj: Převzato od Neuronics (2006).

Manipulátor, na kterém je tato práce vyvíjena, je vybaven univerzálním uchopovačem. V čelistech jsou zabudovány infračervené senzory a lokální snímače síly. Mohou obsahovat i snímače vodivosti (Neuronics, 2006).

## Souřadný systém

Robot Katana může pracovat ve třech souřadných systémech (Neuronics, 2006). Světový souřadný systém (Kw), souřadný systém základny (Kb) nebo souřadný systém nástroje (Ktool)




Obr. 10 Souřadné systémy manipulátoru.
Zdroj: Převzato od Neuronics (2006).

### Použití v úloze hraní šachů

Manipulátor katana je přímo uzpůsoben k práci na sdíleném pracovišti s člověkem. Z hlediska bezpečnosti je to vhodný robot. Dále splňuje požadavek rozhraní v programovacím jazyce C++. Vnitřní reprezentace inverzní kinematiky a metod pro lineární pohyb usnadňuje zadávání koncových souřadnic. Je také možné použít pro vývoj aplikace software Visual Studio 2010.

## 4.1.2 Možné alternativy

### Manipulátor MELFA RV2-AJ

MELFA RV2-AJ je kompaktní průmyslový robot od společnosti Mitsubishi electric. Tento konkrétní model má pět stupňů volnosti tzv. 5 DOF. Je připravený vykonávat různé úlohy. Například odstraňování či ukládání malých prefabrikovaných částí výrobku, zjišťování kvality výrobků, manipulace se vzorky v lékařských laboratořích a mnoho dalších. Může být vybaven jedním elektrickým uchopovačem, nebo až dvěma pneumatickými. Na uchopovací části je vybaven pneumatickými přípojkami pro další nástroje (Mitsubishi, 2010).

### Možnosti programování

Robot MELFA je možno naprogramovat pomocí robotického jazyka MELFA BASIC IV od společnosti Mitsubishi electric. Při spuštění programu je možné sledovat pohyb a graficky jej zobrazovat v SW RT ToolBox2. Komunikaci s vnějším světem zajišťují vstupně výstupní digitální porty (Melsoft, 2013).


Obr. 11 Manipulační ruka Melfa RV-2AJ
Zdroj: Převzato od Mitsubishi (2010).

### Použití v úloze hraní šachů

Z důvodu vyššího výkonu a dosahování vyšších rychlostí než robot Katana 300s není vhodný pro přímé sdílení pracovního prostoru s člověkem. Osazení senzory a prostředky pro komunikaci je konstrukčně náročné a nákladné. Proto nebyl vybrán pro realizaci úlohy, kterou se tato práce zabývá. Manipulátor MELFA RV2-AJ je vhodný spíše pro průmyslové nasazení a manipulaci s předměty na výrobní lince.

# 5 Zpracování obrazu

## 5.1 Hardware

Pro snímání obrazu v tomto řešení je využita webová kamera. Výhodami jsou nízká pořizovací cena, snadné ovládání a možnost uchycení na manipulátor. Podmínkou pro použití konkrétní kamery je vybavenost funkcí autofokus. Protože se s kamerou bude pohybovat a mohou se měnit i světelné podmínky, bylo nutné pořídit kameru, která se s těmito změnami vyrovná.

Pro první testování byla k dispozici kamera Genius s funkcí autofokus a rozlišením 640 na 480 obrazových bodů. Automatické ostření na této kameře bylo dostačující. Se světelnými podmínkami se již vyrovnávala hůře. Vyhledávání vzorů v obraze bylo bohužel nespolehlivé. Všech 32 vzorů nalezla průměrně v jednom případě z 20 pokusů.

Byla proto pořízena webová kamera Microsoft LifeCam studio. Tato kamera se dokáže lépe vyrovnat se světelnými podmínkami díky zabudovanému předzpracování obrazu. Výhodou je také možnost snímání v rozlišení 1920 na 1080 obrazových bodů. Tato kamera dokázala při dobrých světelných podmínkách nalézt všech 32 vzorů 19 krát z 20 pokusů. Při horších světelných podmínkách to bylo 15 krát z 20 pokusů. Tyto parametry a testování hledání vzorů prokázaly, že kamera je pro tuto úlohu vhodná.


Obr. 12 Webová kamera Microsof LifeCam studio
Zdroj: Převzato od Limcorp (2013)

## 5.2 Software

### 5.2.1 OpenCV

Open source computer vision and machine learning software library (dále jen „OpenCV“) je knihovna, navržena pro zpracování obrazu, zaměřena na aplikace běžící v reálném čase. Je připravena využívat více jádrové zpracování. Její velkou výhodou je rozšířenost po celém světě. Uživatelská komunita čítá tisíce členů a díky tomu je knihovna OpenCV stále vyvíjena. Rozsah jejího využití sahá od bezpečnostních aplikací až po pokročilou robotiku. Více o možnostech využití knihovny je možné nalézt v OpenCV (2013).

#### Rozhraní

Knihovna OpenCV obsahuje rozhraní v jazyce C++, C, Python a Java a podporuje operační systémy Windows, Linux, Mac OS, iOS a Android. Nativně je knihovna vyvíjena v jazyce C++ (OpenCV, 2013).

#### Použití v úloze hraní šachů

Knihovna OpenCV je v této práci použita z důvodu splnění následujících kritérií.

- Rozhraní programovacího jazyka C++,
- Optimalizované algoritmy na rychlost,
- přímé snímkování z kamery
- připravenost pro Visual Studio 2010.
- Implementace algoritmu Hough Circles

### 5.2.2 Možné alternativy

#### VisionLab

VisionLab je aplikace pro strojové vidění, vyvíjená společností Mitov software. Implementuje mnoho algoritmů pro zpracování obrazu. Umožňuje detekci obličejů, detekci a sledování objektů, detekci hran. Obsahuje algoritmus Hough Circles pro vyhledání kružnic a mnoho dalších. Největší výhodou je malý počet programového kódu, který je třeba pro tvorbu funkčních aplikací (VisionLab, 2013).

Nevýhodou použití v úloze hraní šachů je zejména placená licence této aplikace a fakt, že funguje pouze v systému ControlWeb. Integrování tohoto software do řešení šachového automatu by mohlo způsobit nestabilitu a přílišnou složitost programu.

# 6 Šachové algoritmy

Modul šachového algoritmu řídí tahy počítačového hráče a verifikuje tahy hráče lidského. Práce na návrhu a implementaci šachového algoritmu by vzhledem k velkému množství kvalitních open source řešení byla nesmyslná. V následujících odstavcích jsou popsány šachové algoritmy, které byly uvažovány pro konstrukci automatu. Při vyhledávání vhodného algoritmu byl brán zřetel na konzolové aplikace, připravené na integraci do grafického uživatelského rozhraní.

## 6.1 Huo Chess

Šachový program Huo Chess je navrhnut jako nejmenší algoritmus pro hru šachy. Je implementován v programovacím jazyce C++. I přes svou malou velikost umožňuje nastavení obtížnosti a je plnohodnotným šachovým algoritmem. Je dostupný pod licencí Microsoft Public License (Huo Chess, 2006). Právě pro jeho malou velikost a přehlednost implementace je zvolen pro řízení počítačového hráče pro celý systém šachového automatu.

## 6.2 StockFish

Šachový engine Stockfish je prezentován jako výkonný šachový algoritmus. Přestože je uvolněn pod licencí General Public License (GPLv3), výkonem se vyrovná komerčně dostupným algoritmům jako je například Rybka či Houdini. Jeho nevýhodou jsou poměrně vysoké nároky na výkon hostitelského počítače. Více o tomto algoritmu lze nalézet ve Stockfish (2010).

## 6.3 GNU Chess

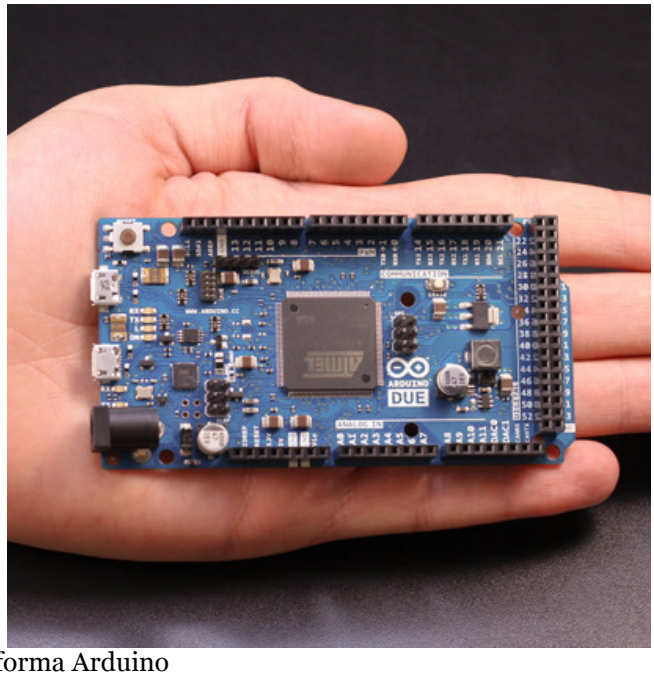
GNU Chess je konzolová aplikace pro řízení počítačového hráče ve hře šachy. V době psaní této práce je k dispozici ve verzi 6.0.3. K dispozici je zdrojový kód v programovacím jazyce C++. První verze tohoto algoritmu vznikla již v roce 1984. Jejím autorem je Stuart Cracraft. Obsahuje možnost nastavení obtížnosti. Nejčastěji bývá použit s grafickým uživatelským rozhraním XBoard nebo glChess. Dostupný je pod licencí General Public License (GPLv3). Více o tomto software lze nalézt v GNU Chess (1996).

# 7 Hardwarové vybavení

## 7.1 Vývojová platforma Arduino

Arduino je elektronická vývojová platforma. Připojení k PC zajišťuje USB port. Deska je osazena vstupy a výstupy pro senzory a akční členy. Mikrokontrolér lze programovat pomocí Arduino programming language (Arduino, 2006).

Platforma může pracovat samostatně nebo komunikovat s počítačem v reálném čase. Jedna z možností je přímá komunikace přes USB port. Zadáním příkazů přímo z počítače lze ovládat vstupní a výstupní digitální porty. Tato komunikace může být použita pro spínání relé, ovládající elektromagnet.




Obr. 13 Vývojová platforma Arduino
Zdroj: Převzato od Arduino (2006)

## 7.2 Prototypová deska a spínací relé

Pro ovládání elektromagnetu při manipulaci figur bylo nutné použít standardní relé ke spínání napájení. Pro propojení elektroniky slouží klasická prototypová deska užívaná pro vývoj elektronických obvodů.

# 8 Vizualizace herní scény

## 8.1 3D vizualizace

Pro vizualizaci 3D objektů je nutné použít grafický engine, který toto zobrazení vygeneruje. Prostředí musí umožnovat komunikaci s řídicím programem. Zobrazení bude reprezentovat rozložení figur na reálné šachovnici. Cyklicky se bude měnit v závislosti na zjištěných změnách herní scény. Vstupní informace pro 3D renderování bude matice řetězců, reprezentující pozice jednotlivých figur. Informace o rozložení bude pro systém zobrazování pouze pro čtení. Možnost ovládání systému přes grafické uživatelské rozhraní je zmíněna v diskuzi.

### 8.1.1 Irrlicht 3D engine

Irrlicht engine je výkonný prostředek pro renderování scény v reálném čase, napsaný v jazyce C++. Je platformě nezávislý. K renderování je možné využít D3D, OpenGL a vlastní softwarový render. Obsahuje vazby na programovací jazyky java, perl, ruby, basic, python. Je pro něj vytvořeno mnoho vzorových ukázek a návodů (Irrlicht, 2013).

Výhodou Irrlicht je velká základna aktivních vývojářů. O kvalitách svědčí mnoho projektů využívajících právě Irrlicht pro generování scény. Irrlicht grafický engine je dostupný zcela zdarma.

### 8.1.2 OGRE 3D engine

OGRE, neboli objektově orientovaný grafický renderovací engine, je flexibilní engine vyvinutý v jazyce C++. Je navrhnutý jako jednoduchý a intuitivní prostředek pro vývojáře 3D grafických aplikací. Knihovna tříd abstrahuje všechny detaily nižších systémových knihoven jako například D3D a OpenGL a poskytuje rozhraní založené na intuitivních třídách (Ogre, 2000).

K dispozici je propracovaná dokumentace a návody pro seznámení. Použití OGRE engine je podmíněno licencí MIT. Při dodržení podmínek copyrightu jej lze pro vývoj aplikací použít bezplatně.

# 9 Realizace šachového automatu

## 9.1 Fyzické upevnění kamery na manipulátor Katana

Manipulátor je osazen jednou webovou kamerou na svém pohyblivém rameni. Pro upevnění kamery na manipulátor jsou použity ocelové objímky se závitem a gumovým těsněním. Díky šroubovému uchycení objímek je možné natočit kameru do požadovaného úhlu. Gumové těsnění drží šroubové spoje na místě a zaručuje neměnnou pozici kamery vůči manipulátoru. Kamera je upevněna přímo na manipulátoru z důvodu zjednodušení a zvýšení spolehlivosti. Při použití externího stojanu na kameru by hrozila kolize. Úloha je díky tomuto řešení lépe přenositelná a kompaktnější. Tato výhoda se projeví zejména při převozu na propagační akce.

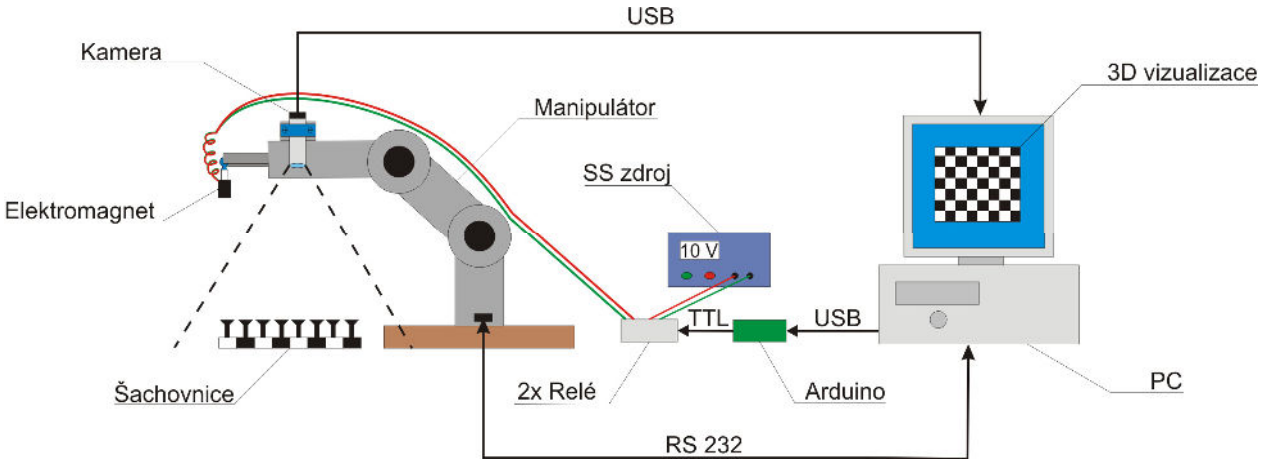
Osazená webová kamera je od společnosti Microsoft. Jedná se o typ LifeCam Cinema. Její největší výhodou je možnost snímání obrazu ve vysokém rozlišení HD rychlostí až 30 snímků za sekundu. Webové kamery s nižším rozlišením vykazovaly nadměrnou nespolehlivost při vyhledávání vzorů v obraze. Kamera má také automatické ostření a funkce, které zajištují ostrý obraz i při detailních záběrech. Její válcovitý tvar umožnil snadné upevnění kolmo na herní scénu, což výrazně zpřesnilo výsledky získávané ze snímaného obrazu.

Obr. 14 Upevnění kamery na manipulátor.

## 9.2 Popis principu celého řešení

Kamera je uchycena na koncovém rameni manipulátoru. Při určené pozici její zorný úhel dovoluje snímat celý herní prostor. Kamera je připojena pomocí USB. Pro manipulaci s figurami je na manipulátoru upevněn elektromagnet, působící svým magnetickým polem na figury. Ovládání elektromagnetu je realizováno dvěma relé, o jejichž spínání se stará vývojová platforma Arduino standardem TTL. Komunikace mezi vývojovou platformou a šachovým počítačem je realizována pomocí USB. Napájení elektromagnetu zajišťuje stejnosměrný stabilizovaný zdroj.

Celkový přehled zapojení všech fyzických komponent včetně typu komunikace je uveden na obr. č. 15 Všechny dílčí součásti jsou dále v tomto textu detailně popsány.

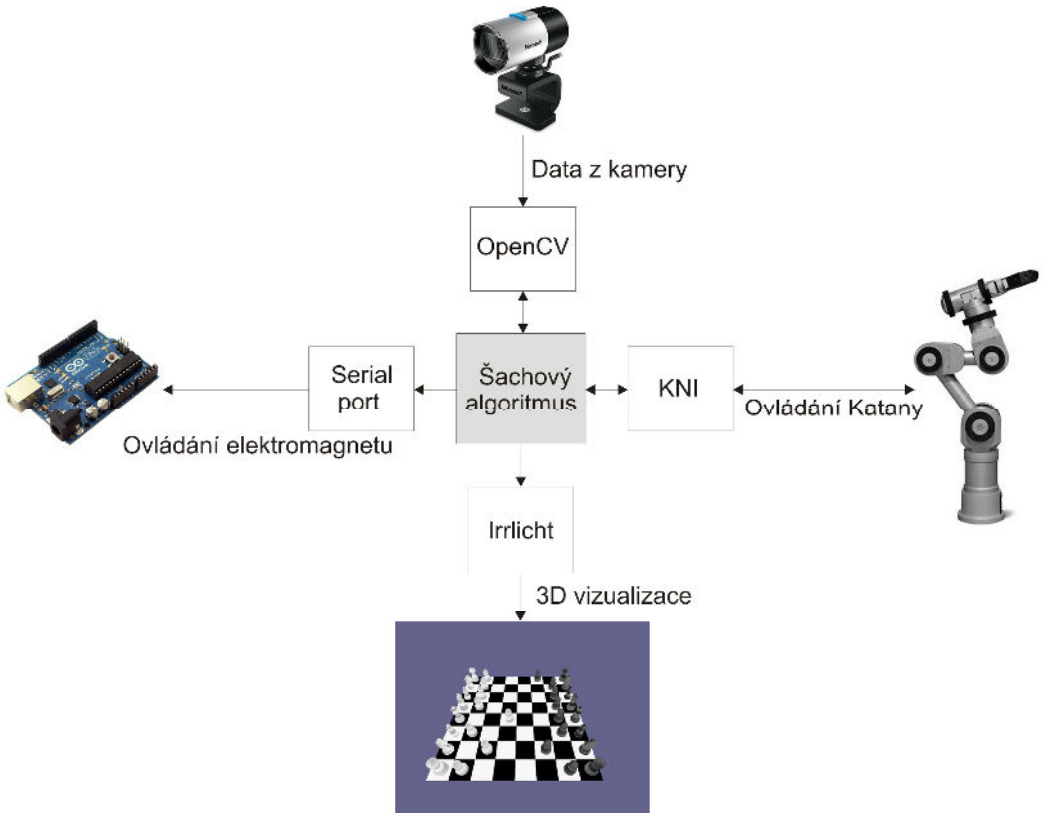



Obr. 15 Schématické znázornění celého řešení šachového automatu

## 9.3 Propojení SW pro snímání obrazu se systémem pohybu a systémem vizualizace

### 9.3.1 Popis modulů programu

Celý systém šachového automatu se skládá z pěti modulů. Modul pro zpracování obrazu, modul pro řízení manipulátoru, modul pro 3D vizualizaci, modul šachového algoritmu a modul pro ovládání elektromagnetu. Všechny použité součásti jsou naprogramovány v programovacím jazyce C++. Pro vývoj programu bylo využito prostředí Microsoft Visual Studio 2010.




Obr. 16 Schématické znázornění jednotlivých modulů šachového automatu

Modul pro ovládání manipulátoru, Katana Native Interface byl připraven pro Microsoft Visual Studio 2008. Automatická konverze, obsažená v novější verzi Visual Studia, se nezdařila. Bylo nutné ručně upravit soubory a změnit jejich metadata. Poté se projekt KNI podařilo přeložit a uvést do provozu.

Modul pro zpracování obrazu se po zahrnutí knihoven OpenCV a správnému nastavení všech závislostí podařilo integrovat. V této fázi vznikl vzorový projekt pro výuku. Bezpečnostní pravidla byla nastavena a mohl se tedy oddělit jako vedlejší produkt vývoje. Podrobný popis zpracování obrazu je uveden dále v textu.

Jako šachový algoritmus byl použit Huo Chess. Stejně jako předchozí součásti je vyvíjen v jazyce C++. Hlavní smyčka programu se odehrává právě v tomto modulu. Generuje výstupy a ošetřuje vstupní informace.

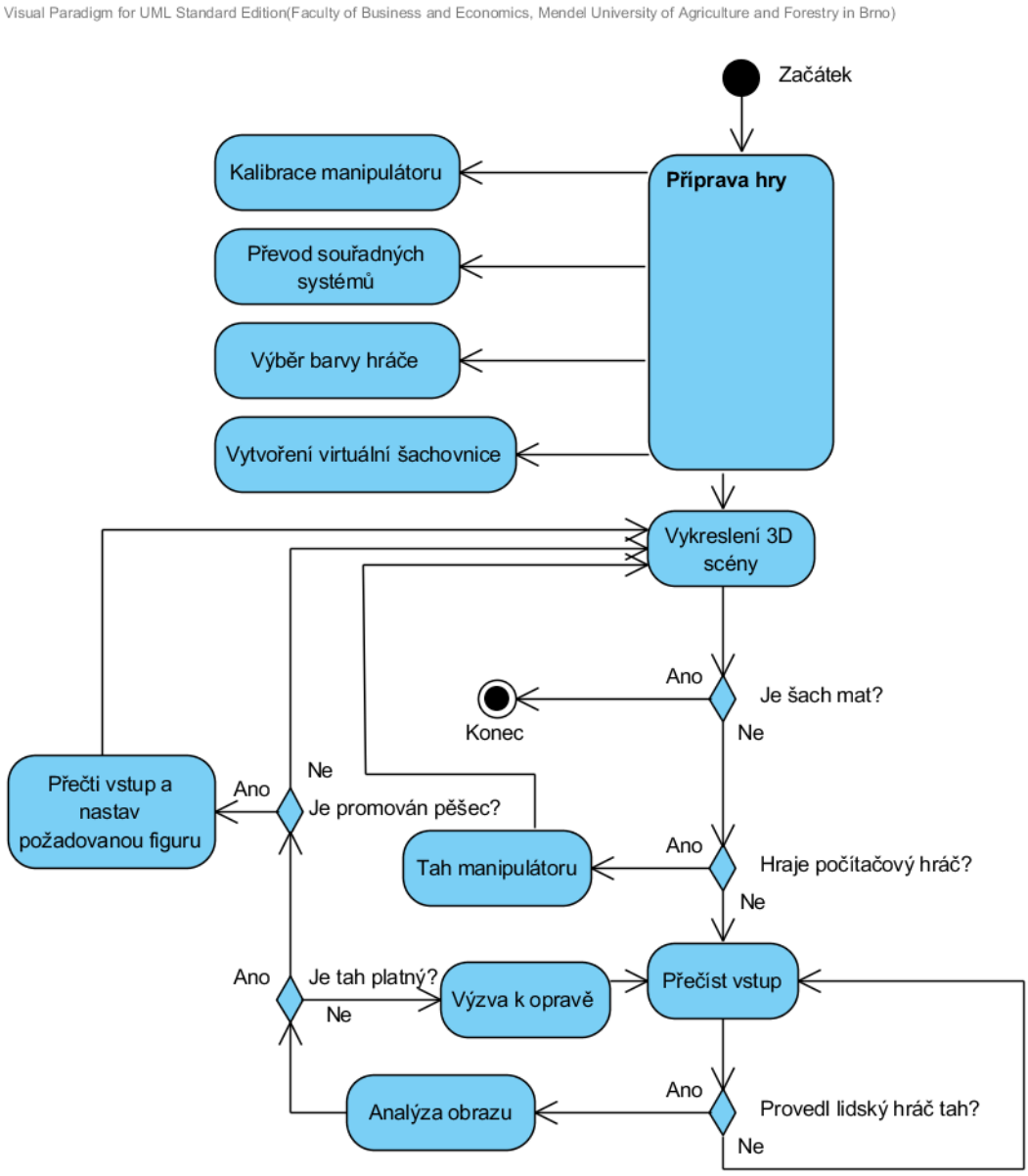
Pro ovládání elektromagnetu slouží vývojová platforma Arduino. V této platformě běží smyčka pro detekci sériové komunikace, přicházející z PC. Právě zasílání příkazů pro ovládání elektromagnetu je hlavní funkcí tohoto modulu. Využívá se nativních knihoven jazyka C++ pro sériovou komunikaci. Podrobnější popis je uveden dále v této práci.

Modul pro vizualizaci herní scény je založen na grafickém enginu Irrlicht. Po každém tahu, lidského i počítačového hráče, vyrenderuje nové pozice figur na šachovnici. Pro správnou funkčnost je nutné, aby byl tento grafický engine spuštěn v druhém vláknu aplikace.

### 9.3.2 Běh programu

Po startu programu je inicializována Katana a poslána instrukce ke kalibraci. Kalibrace manipulátoru je nutná po každém odpojení napájení. Pokud připojení i kalibrace proběhne v pořádku, manipulátor se přesune do přednastavené pozice pro snímání obrazu. Po přípravě kalibračních bodů dojde k výpočtu hodnot, určujících převod souřadných systémů. Dalším krokem je vytvoření virtuální šachovnice, kde jsou uloženy souřadnice všech čtverců. Podrobný tohoto kroku je dále v tomto textu.

Po těchto krocích je program připraven na hru. Otevře se grafické okno a zobrazí se 3D reprezentace šachovnice s figurami na startovních pozicích. Uživatel je vyzván k volbě barvy, za kterou bude hrát. V případě výběru bílých figur je vyzván k provedení prvního tahu. V opačném případě provede tah manipulátor na základě instrukce od šachového algoritmu. Po každém tahu se vykreslí nové rozložení na 3D zobrazení šachovnice a probíhá kontrola, zda není konec hry. Pokud má jeden z hráčů šach mat, hra končí. V opačném případě se hráči střídají ve hře. Po tahu počítačového hráče se pouze překreslí 3D zobrazení. Nedochází k žádné analýze obrazu, protože algoritmus sám zavede tah do vnitřní reprezentace rozložení figur. Zde je v programu slabé místo. Pokud by došlo k chybě při manipulaci a figura nezapadla na očekávané místo, musí zasáhnout operátor a tah dokončit, jinak není možné pokračovat. Pokud je na řadě lidský hráč, je vyzván k tahu. Na obrazovce se vypíše, kterou klávesu má stisknout pro potvrzení tahu. Po stisku požadované klávesy dojde k analýze obrazu. Ve snímku z kamery se vyhledají vzory a podle změn na hrací desce se určí tah hráče. Informace o nových pozicích figur je otestována na platnost tahu. Pokud šachový algoritmus určí tah jako neplatný, lidský hráč je vyzván k opravě tahu. Platný tah je dále testován na promoce pěšce. Pokud lidský hráč dosáhne pěšcem opačný konec šachovnice, může si zvolit, jakou figuru bude pěšec dále ve hře reprezentovat. Na obrazovce se zobrazí čtyři možnosti pod číslicemi 1 až 4. Hráč si může zvolit dámu, věž, jezdce nebo střelce. Po zadání volby se tato proměna projeví jak v 3D grafickém zobrazení, tak ve hře. V této smyčce hra probíhá až do konce. Po ukončení hry je hráči zobrazen výsledek utkání.




Obr. 17 Diagram aktivit programu šachového automatu

## 9.4 Kalibrace kamery

Před prací s daty, získanými z webové kamery, je nutné zaměřit se na kalibraci obrazu. Při snímání obrazu dochází u některých čoček, zvláště v případě webových kamer, ke zkreslení, způsobeném zakřivením čočky. Při přepočítání jednotlivých pixelů na reálné jednotky vzdálenosti bez předchozí kalibrace vzniká chyba.

V případě soudkovitého zkreslení, odpovídající namapování obrazu na imaginární soudek, vzniká zkreslení, které se zvětšuje se vzdáleností od optického stře-

du. Tangenciální zkreslení vzniká tehdy, když není snímaná rovina naprosto rovnoběžná s rovinou kamery.

Obě tato zkreslení lze minimalizovat kalibrací. OpenCV nabízí možnost upravit obraz pomocí snímání obrazců se známými parametry (OpenCV, 2011). Nejčastějším obrazcem pro kalibraci je černobílá šachovnice. Jako vstupní parametry se udává počet vnitřních hran na výšku a na šířku šachovnice a rozměr čtverců. Dále je možné jako obrazce použít kruhy se známou vzdáleností středů. Metoda funguje na principu nalezení obrazce ve snímku a přemapování pixelů podle vzorců tak, aby snímek odpovídal realitě. Vzorce (1) a (2) slouží pro radiální kalibraci (OpenCV, 2011).

$$x_{corrected} = x\left(1 + k_1 r^2 + k_2 r^4 + k_3 r^6\right) \tag{1}$$

$$y_{corrected} = y\left(1 + k_1 r^2 + k_2 r^4 + k_3 r^6\right) \tag{2}$$

Vzorce (3) a (4) umožní kalibraci tangenciální (OpenCV, 2011).

$$x_{corrected} = x + \left[2 p_1 xy + p_2\left(r^2 + 2x^2\right)\right] \tag{3}$$

$$y_{corrected} = y + \left[p_1\left(r^2 + 2y^2\right) + 2 p_2 xy\right] \tag{4}$$

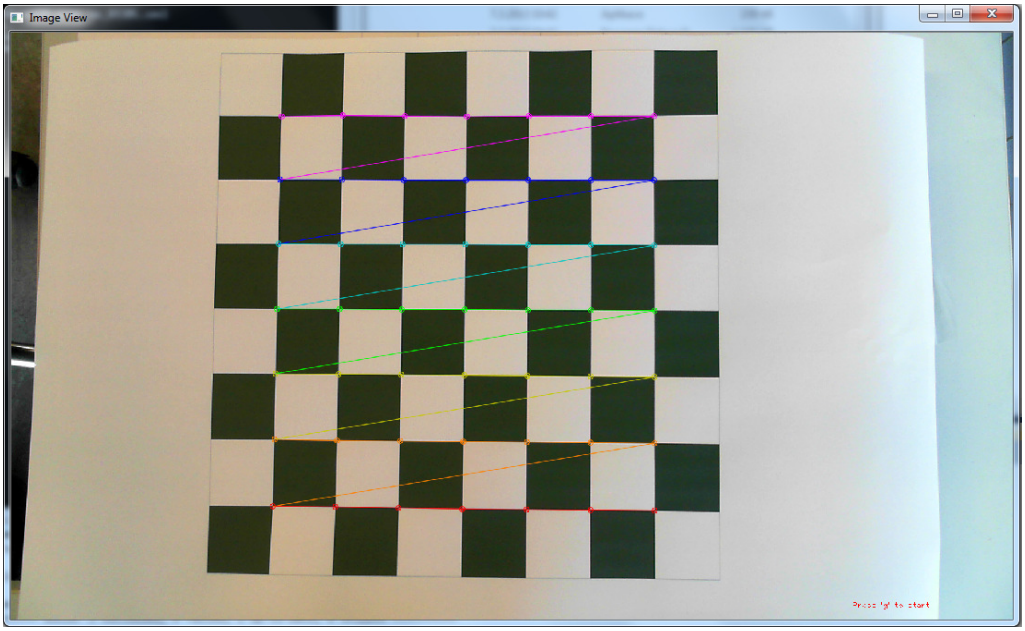
Získáme parametry zkreslení viz vzorec (5) (OpenCV, 2011).

$$Distortion_{coeficients} = \left(k_1 k_2 p_1 p_2 k_3\right) \tag{5}$$

Po získání těchto parametrů je možné provést vlastní výpočet převodu souřadnic obrazu. Výpočet je znázorněn vzorcem (6) (OpenCV, 2011).

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \tag{6}$$

Uvedení $w$ ve vzorci (6) je z důvodu použití homografického souřadného systému kde $w = Z$. Neznámé parametry jsou $f_x$ a $f_y$ (Ohniskové vzdálenosti kamery) a $\left(c_x, c_y\right)$ optické středy vyjádřené v pixelových souřadnicích. Pokud je pro obě osy použita společná ohnisková vzdálenost $a$ (obvykle zvoleno 1) lze použít vztah $f_y = f_x * a$ a získat jednu ohniskovou vzdálenost $f$. Matice obsahující tyto čtyři parametry je označována jako matice kamery. Koeficienty zkreslení jsou nezávislé na rozlišení kamery. Více o kalibraci lze nalézt v OpenCV (2011). Tato kalibrace umožní přepočítat obrazové body na jednotky vzdálenosti reálného světa v systému řízení manipulátoru.




Obr. 18 Kalibrace kamery pomocí šachovnice.

## 9.5 Vyhledání vzorů v obraze

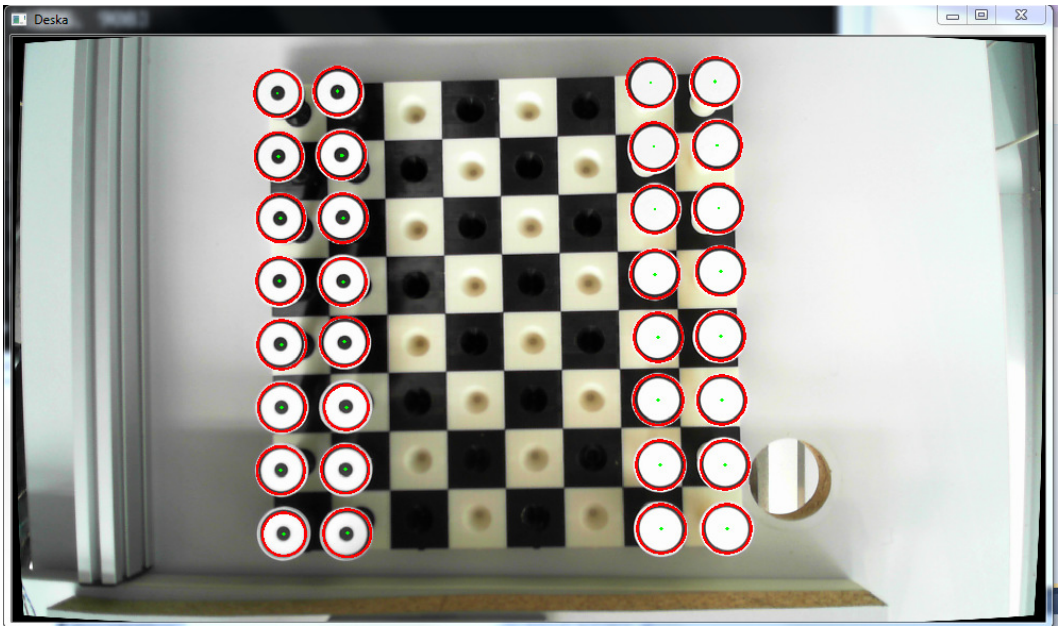
Stěžejní vzor pro vyhledávání objektů v obraze je v této úloze kružnice. Pomocí metody knihovny OpenCV *HoughCircles(),* zmíněné výše v této práci, lze tyto kružnice vyhledat. Před vyhledáváním vzorů v obraze je třeba nejprve provést zpracování snímku. Pro převod snímku do odstínu šedé je použita metoda *cvtColor()*. Vstupním parametrem je zdrojový snímek a typ převodu. Konstanta pro převod je v tomto případě *CV_BGR2GRAY* .Výstupem je cílový snímek v odstínech šedé.

Dalším krokem je vyprahování snímku. Toho lze dosáhnout metodou *treshold()*. Nejdůležitějším parametrem této metody je práh. Při správném nastavení prahu se zvýrazní kontury kružnic. Hodnota prahu se volí v rozmezí od 0 do 255. Z hlediska hodnocení spolehlivosti ve vyhledávání kružnic nejlepší výsledek vykazovala metoda s hodnotou prahu nastavenou na 100.

Dále je nutné snímek rozmazat, aby se předešlo falešné detekci kružnic. Tento problém *řeší metoda GaussianBlur()*. Vstupním parametrem je zdrojový snímek, velikost okolí a standardní odchylka x a y. Výstupem je rozmazaný snímek. Všechny tyto metody upravily snímek tak, aby vyhledání kružnic bylo co nejspolehlivější i za snížených světelných podmínek a dalších nepříznivých jevů.

Jako hlavní metoda pro vyhledání kružnic je použita *HoughCircles()*. Vstupní parametry jsou zdrojový snímek, vektor kružnic, metoda vyhledávání, poměr rozlišení, minimální vzdálenost mezi středy, maximální a minimální průměr kružnic. Jako metoda vyhledávání je použita *CV_HOUGH_GRADIENT*. Poměr rozlišení s hodnotou 1. Vzdálenosti středů a velikost kružnic je dynamicky odvozena od veli-

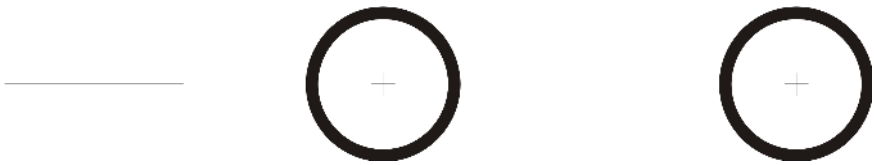
kosti zdrojového snímku. Výstupem je vektor kružnic, z kterého můžeme zároveň získat jejich středy.




Obr. 19 Nalezení vzorů při inicializaci hry.

## 9.6 Převod souřadných systémů

Pro identifikaci polohy objektů v obraze je použit přepočet z obrazových bodů do jednotek vzdálenosti v souřadném systému manipulátoru Katana. Protože je obraz zbaven zkreslení a je známa výška figur, je možné zjistit poměr přepočtu bodů dvěma kružnicemi. Střed obou kružnic je nastaven kolmo na osu $x$ v souřadném systému manipulátoru. Vzdálenost středů kružnic, jakožto i hodnota na ose y v souřadném systému manipulátoru, jsou známé. Jednoduchým výpočtem lze zjistit posunutí na ose $x$ a $y$ v souřadném systému manipulátoru vůči bodům v obraze. Také lze vypočítat převodní poměr mezi obrazovými body a jednotkami vzdálenosti cílového souřadného systému. Výpočet proběhne podle vzorce 1.



Obr. 20 Kružnice pro převod souřadných systémů.

Souřadnice manipulátoru, které odpovídají umístění kalibračních kruhů, jsou dány. Pro tento případ je hodnota na ose $x = 0$ pro oba kruhy, na ose $y = 180$ pro jeden a $y = 280$ pro druhý kruh. Osa $z$ je dána výškou figur a výškou šachovnice. Po zpracování obrazu a nalezení kruhů získáme jejich souřadnice v obrazových bodech ve snímku. Rozsah těchto souřadnic je dán rozlišením snímku. V tomto případě 1920 na 1080 obrazových bodů. Pro výpočty posunutí jednotlivých os je třeba určit převodní poměr mezi obrazovými body a milimetry. Pro jednoduchý výpočet využijeme vzdálenost mezi kruhy, která je přesně 100 milimetrů. Určení proběhne pomocí vzorce (7). Výsledkem je koeficient pro převod.

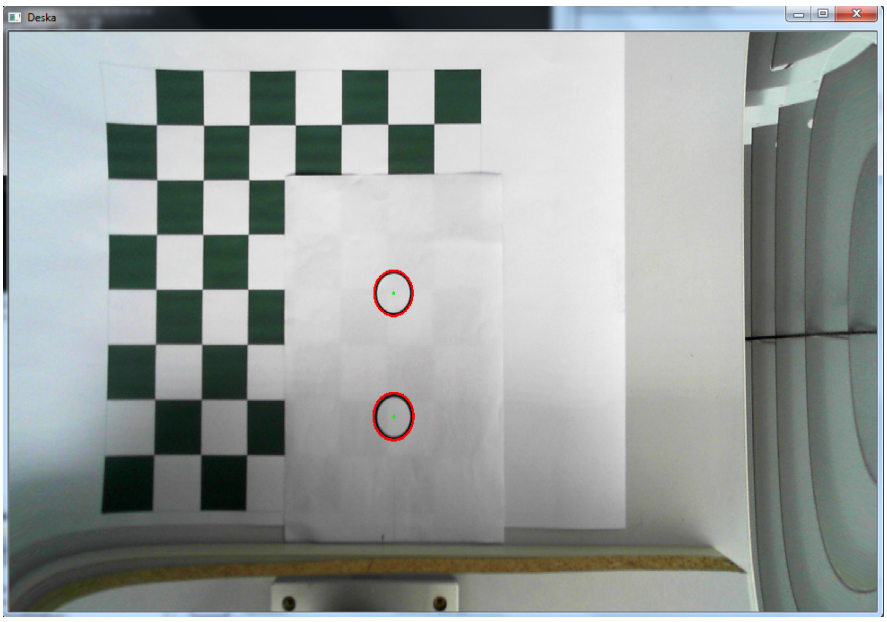
$$PxToMm_{koeficient} = \left| y_{kruh1} - y_{kruh2} \right| / 100 \tag{7}$$

Pomocí zjištěného koeficientu převodu lze dopočítat posunutí v jednotlivých osách. Posunutí v ose $x$ je určeno vzorcem (8). Vzorec pro výpočet posunutí v ose $y$ musí obsahovat obrácení souřadného systému (9).

$$x_{koeficient} = 0 - \left( x_{kruh1} / PxToMm_{koeficient} \right) \tag{8}$$

$$y_{koeficient} = 180 - \left( (1080 - y_{kruh1}) / PxToMm_{koeficient} \right) \tag{9}$$

Vypočítané koeficienty převodu jsou použity k převodu souřadných systémů.

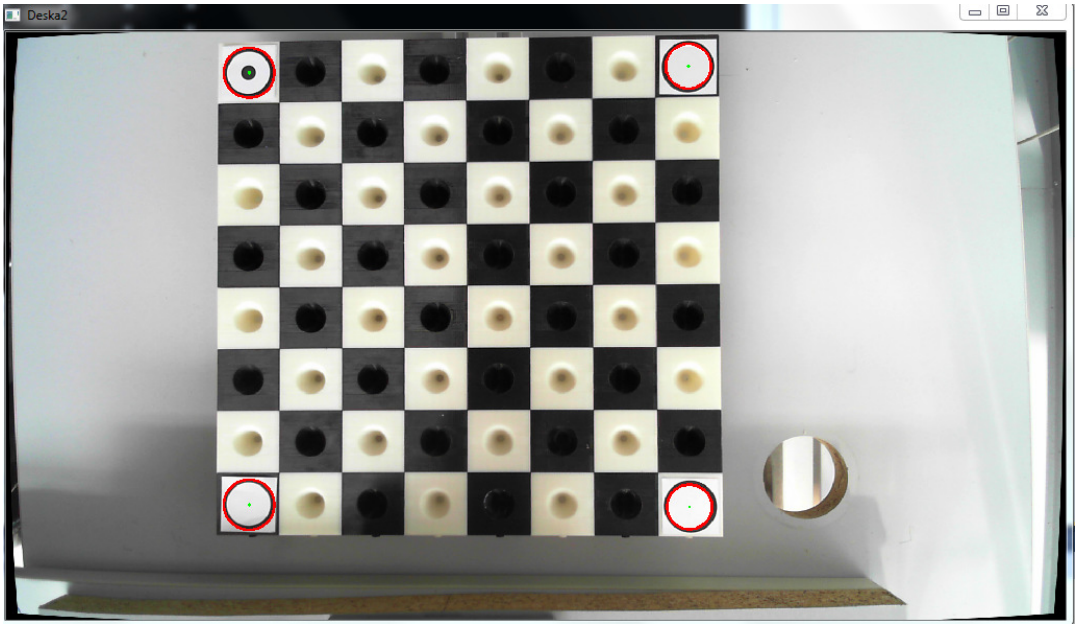



Obr. 21 Zobrazení nalezených kružnic a jejich středů.

Koeficienty převodu souřadných systému je nutné zjistit ve dvou úrovních. První úroveň odpovídá výšce figur. Druhá úroveň je shodná s výškou šachovnice. Tímto postupem je zajištěno přesné polohování robotu na obě úrovně modelu i při použití pouze jedné kamery pro snímání herní scény.

## 9.7 Snímání herní šachovnice

Při vytváření vnitřní reprezentace polí šachovnice dochází k výpočtu všech souřadnic ze čtyř referenčních bodů v rozích šachovnice. Metoda nalezne ve snímku čtyři kruhy a jejich středy. Určí, který roh je levý horní, levý dolní, pravý horní a pravý dolní. Protože se šachovnice skládá z 64 identických polí, lze pomocí interpolace dopočítat pozice jednotlivých čtverců. Aby se zmenšila chyba, je hodnota vzdáleností protilehlých rohů zprůměrována. Z interpolace získáme délku strany a souřadnice levého horního rohu každého čtverce. Na základě těchto zjištěných hodnot je vytvořena matice, v níž je vytvořeno 64 čtverců pomocí instancí třídy *Rect*. Na tyto objekty lze poté aplikovat metodu *Contains()*, díky které lze určit příslušnost bodu, reprezentujícího figuru, do příslušného čtverce, reprezentující pozici na šachovnici. Tato informace je použita při tahu manipulátoru, nesoucího figuru na prázdné cílové pole.




Obr. 22 Nalezení rohů šachovnice

## 9.8 Počáteční inicializace figur

Při prvním snímání všech figur na startovní pozici se vytvoří vnitřní reprezentace šachovnice ve výšce figur. Z důvodu perspektivního zkreslení se tato reprezentace

liší od pozic herní desky. Současně se naplní vnitřní matice pozic s rozlišením na černé a bílé figury.

Algoritmus prohledává jednotlivé snímky, dokud nenajde všech 32 figur. Nalezené kružnice uloží do vektoru. Podle zjištěných souřadnic středů kružnic $x$ a $y$ zjistí rohové figury. Protože standardní šachovnice má známé rozměry 8 na 8 čtverců, vzdálenost dolních a horních středů od sebe, po vydělení číslem 7, udá délku hrany virtuálního čtverce. Po odečtení poloviny hrany čtverce od souřadnic levého horního rohu je znám počátek souřadného systému šachovnice, začínajícím v jejím levém horním rohu. Pomocí struktury *Rect*, obsažené v OpenCV, jsou vytvořeny čtverce, reprezentující oblasti, kde se mohou vyskytovat figury na jednotlivých souřadnicích. Všechny tyto operace pracují pouze se souřadnicemi snímku. Po této základní inicializaci virtuální šachovnice následuje určení bílých a černých figur.

### 9.8.1 Rozlišení černých a bílých figur

Počáteční pozice figur na hrací ploše je známá. Pro určení pohybu figur na šachovnici a zadání tahu člověka do šachového algoritmu postačuje rozlišení figur na černé a bílé.

Algoritmus vyhledá na snímku 32 vzorů. Pro každý nalezený vzor vytvoří výřez s rozměry nalezeného vzoru a uloží do vektoru. V dalším kroku zjistí, pomocí virtuální šachovnice, na jaké pozici se nacházejí jednotlivé vzory. Pro nalezení shody, zda střed nalezeného vzoru je uvnitř čtverce na konkrétní souřadnici je použita metoda *contains()*. Vstupními parametry jsou čtverec *Rect* a bod *Point*. Výstupem je hodnota pravda, pokud je bod uvnitř čtverce a nepravda, pokud se uvnitř nenachází. Po určení pozice vzoru na konkrétní čtverec se určí, zda se jedná o černou nebo bílou figuru. Vzor pro černé figury má na středu menší kruh. Z pohledu spolehlivosti určení druhu vzoru se neosvědčily metody *HoughCircles()* pro vyhledání kruhu ani *findContours()* pro vyhledání kontur. Konečným řešením je průchod všemi body výřezu po vyprahování a určení procentuální zastoupení černých a bílých bodů. Protože jsou rozměry výřezu malé, nedošlo k žádnému znatelnému časovému zpoždění při určení. Po této úpravě již nedošlo k chybnému určení příslušnosti figur k bílé nebo černé barvě.

## 9.9 Zjištění pozic figur po tahu živého hráče

### 9.9.1 Kontrola počtu nalezených figur

Při vyhodnocování šachovnice po tahu živého hráče je určen počet zjištěných vzorů, reprezentujících figury. Množství zjištěných vzorů musí odpovídat počtu figur z minulého regulérního tahu, nebo o jednu méně, pokud v tomto konkrétním tahu došlo k vyhození jedné protivníkovy figury. Pokud počet zjištěných vzorů neodpovídá, kamera automaticky sejme nový snímek a analýza znovu vyhodnotí obraz. Díky tomuto omezení se automaticky zahodí všechny snímky, ve kterých není dostatečný počet figur pro další smysluplné zpracování. Zejména případy, kdy kame-

ra chybně vyhodnotí vzory v obrazu. Pokud počet figur vyhovuje omezení, jsou souřadnice nalezených vzorů uloženy do dynamické struktury.

### 9.9.2 Určení pozice figur

Dalším krokem v programu je určení polohy figur na šachovnici. Přiřazení na konkrétní souřadnici šachovnice je provedeno pomocí virtuální šachovnice, vytvořené v počáteční inicializaci. Virtuální šachovnice obsahuje na jednotlivých polích strukturu *Rect*, která reprezentuje hranice čtverce reálné šachovnice. Porovnáním souřadnic zjištěných vzorů a čtverců získáme aktuální rozložení figur na šachovnici. Současně je určeno, zda se na poli nalézá černá nebo bílá figura.

Tabulka 2 příklad rozložení figur ve vnitřní reprezentaci

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | 0 | 2 | 0 | 0 | 2 |
| 1 | 0 | 1 | 0 | 0 | 0 | 2 | 2 |
| 1 | 1 | 0 | 0 | 0 | 0 | 2 | 2 |
| 1 | 0 | 0 | 1 | 0 | 2 | 0 | 2 |
| 1 | 1 | 0 | 0 | 0 | 0 | 2 | 2 |
| 1 | 1 | 0 | 0 | 0 | 0 | 2 | 2 |
| 1 | 1 | 0 | 0 | 0 | 0 | 2 | 2 |
| 1 | 0 | 0 | 1 | 0 | 0 | 2 | 2 |

### 9.9.3 Kontrola počtu změn v poloze figur

Vygenerovaná tabulka s pozicemi figur je porovnána s tabulkou z minulého tahu. Porovnáním těchto dvou tabulek zjistíme počet změn v rozložení, jež je dalším omezujícím faktorem, pro zvýšení spolehlivosti. Jediné přípustné hodnoty změn jsou dvě a čtyři. Pokud počet změn na virtuálních šachovnicích není vyhovující, kamera sejme nový snímek a analýza probíhá znovu od začátku. Tento zdánlivě složitý proces je nutné provést pro zajištění vysoké spolehlivosti zadání tahu.

Při počtu dvou změn v porovnání tabulek je nezbytné zjistit výchozí souřadnici tahu. Číslo nula indikuje výchozí pole tahu v případě vyhození protivníkovi figury i při jednoduchém tahu. Druhé pole, kde došlo ke změně hodnoty je tedy automaticky cílové. Následuje převedení souřadnic tabulky do souřadnic herní desky šachového algoritmu a zadání tahu. Souřadnice herní desky jsou do algoritmu zadávány ve formátu „A-H“ pro řádek a „1-8“ pro sloupec.

Při počtu čtyřech změn je provedena kontrola, zda lidský hráč neprovedl tzv. velkou nebo malou rošádu. Velká a malá rošáda je jediný tah, ve kterém se pohybují dvě figury stejné barvy. Kontrola je provedena jednoduše tak, že se kontroluje, na kterých polích vznikla změna a zda odpovídají pozicím před a po provedení velké nebo malé rošády. Konkrétní rozestavění figur při provádění rošády vychází z obecných pravidel hry šachy a nebude zde popisováno. V případě, že tah odpovídá těmto změnám, je zadán pohyb krále o dvě pole na cílovou souřadnici, což je v šachovém algoritmu chápáno, jako provedení rošády.

### 9.9.4 Neplatný tah

Pokud lidský hráč provede neplatný tah, je na to upozorněn a požádán o opravu. Tah, i když je neplatný, projde analýzou obrazu a je zadán do šachového algoritmu. Šachový algoritmus dokáže určit neplatný tah a upozorní na něj. Neplatný tah je detekován i v případě, kdy se lidský hráč pokusí nesprávně vyhodit figuru počítačového hráče. Při vyhodnocení tahu jako neplatného, se automatické snímání fyzické šachovnice zastaví a program čeká na potvrzení od lidského hráče, že provedl regulérní tah. Všechny změny ve virtuální šachovnici, které proběhly v rámci analýzy obrazu a počtu figur se zahodí a tento tah není zaznamenán do vnitřní reprezentace. Pokud by k tomuto zahození nedošlo, následující tah by nemusel být vyhodnocen správně. Po opravě chyby tahu program pokračuje standardně.

## 9.10 Vytvoření vzorového projektu pro výuku

Vzorový projekt pro výuku obsahuje systém pro řízení manipulátoru a systém pro vyhledání vzoru v obraze. Projekt je realizován v prostředí Microsoft Visual Studio 2010. V systému pro zpracování obrazu je použita metoda pro kalibraci kamery a převod souřadných systémů. Dále metoda pro vyhledání vzorů v obraze.

Systém řízení manipulátoru je zejména upraven tak, aby manipulace byla bezpečná. Pokud by manipulátor provedl akci, která by vedla k jeho poškození nebo k poškození jeho okolí, motory se zablokují. Pouze signál pro odblokování motorů dovolí programátorovi (studentovi) pokračovat v práci.

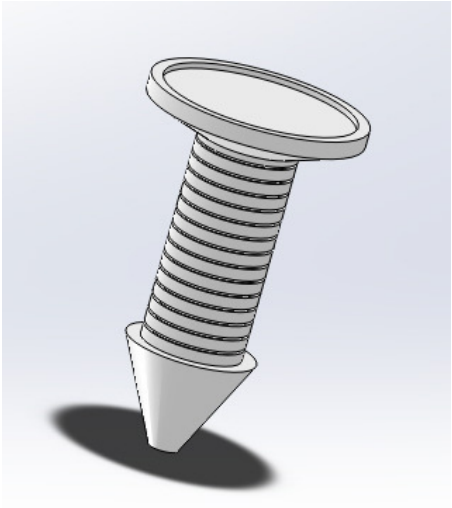
Projekt slouží jako šablona pro studentské práce. Základní funkcionalitu lze rozšířit o manipulaci s nalezenými předměty, jejich přesun na označené pozice, skládání na sebe a mnoho dalších.

## 9.11 Modelování a tisk šachovnice a figur na 3D tiskárně

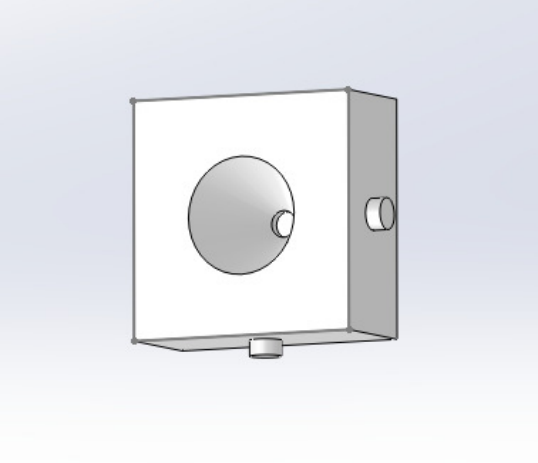
Šachové figury, které jsou standardně používány pro hru šachy, nebyly pro tento typ úlohy vhodné. Objekty, se kterými manipulátor pohybuje, musí mít na vrcholku plochu, ve které je zabudována kovová deska pro uchopení pomocí elektromagnetu pokrytá vzorem pro rozpoznání kamerou. Dále je spodní část modelovaných figur vytvarována do tvaru kuželu. Od středu kuželu je vedena dutina pro vložení železné osy o tloušťce 5 mm. Tento kužel zapadne do připravené výhlubně a figura se silou gravitace vycentruje na požadované souřadnici. Všechny tyto prvky figur zvyšují spolehlivost celé úlohy.
Vedlejším cílem je také prezentace 3D tiskárny. Pro vymodelování figur byl použit software SolidWorks. Jeho výhodou je přímý export modelu do stereolitografického formátu tzv. STL, který 3D tiskárna od společnosti Dimension vyžaduje jako vstupní formát.

Podložka šachovnice je složena z černých a bílých podstavců, ve kterých je kuželová výhlubeň. Výhlubeň rozměrově odpovídá kuželu na spodní straně figur. Při usazení figury na její nové souřadnice zapadne kužel do výhlubně a dojde tak k vycentrování figury. Podstavce jsou opatřeny zámky, které umožní jejich přesné

složení do jedné podložky. Díky možnosti rozložení je celá úloha lehce transportovatelná. Barevně jsou figury i podložky rozlišeny přímo materiálem, ze kterého jsou vytištěny.



Obr. 23 3D model figury věž.



Obr. 24 3D model jedné komponenty šachovnice.

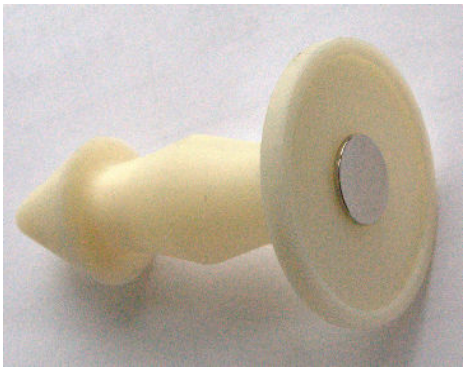
## 9.12 Fyzická úprava šachovnice a figur

### 9.12.1 Vytvoření šachovnice

Po tisku je nutné komponenty upravit tak, aby je bylo možné použít pro konstrukci úlohy. Součásti šachovnice jsou vytištěny plastem ABS již v požadovaných dvou barvách. Povrchová úprava tedy není nutná. Komponenty šachovnice jsou navrženy tak, aby se po spojení vycentrovaly pomocí kolíků a děr po stranách. Díky použití spojovacího média drží šachovnice tvar i při manipulaci.

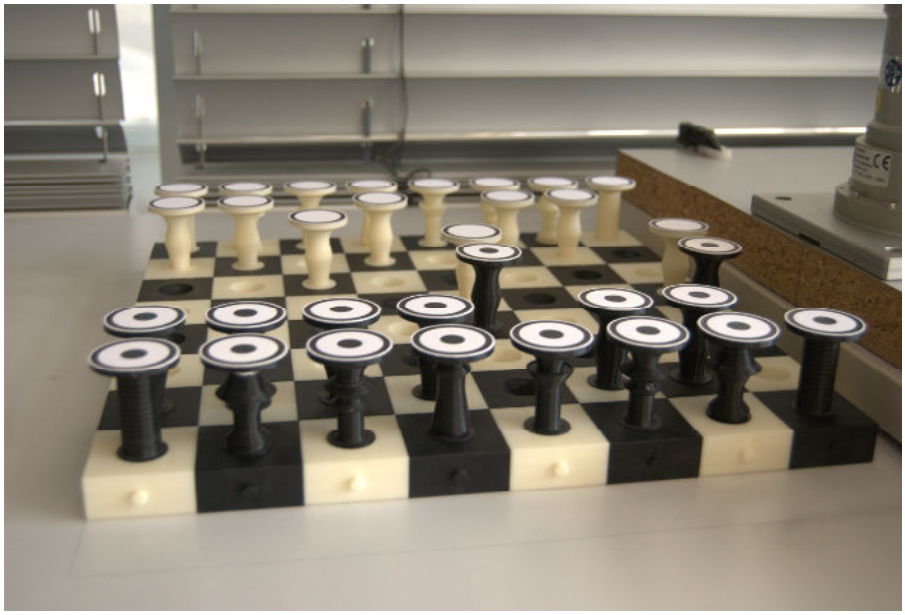
### 9.12.2 Úprava figur

Pro použití v úloze hraní šachů musí být jednotlivé figury opatřeny magnety na své horní straně. Všechny magnety jsou umístěny tak, aby jejich póly směrovaly stejným směrem vzhledem k dané figuře. Do figur jsou použity neodymové magnety s šířkou dvanáct milimetrů, výškou jeden milimetr a magnetickou silou přibližně šest set gramů. Magnety mají severní a jižní pól na rovných kruhových plochách. Tyto magnety jsou vhodné do aplikací elektromechanického charakteru. Díky použití magnetu jako média pro uchopení, namísto kovového kotouče, se mnohonásobně zvýší spolehlivost celé úlohy. Hlavní výhodou je vycentrování celé figury při uchycení elektromagnetem. Dále je na spodní straně figury prostor pro uchycení kovového válce pro přesun těžiště.



Obr. 25 Umístění magnetu na figuru.

Nakonec jsou figury opatřeny vzorem pro rozpoznání v obraze. Vzor je umístěn na horní část figury a překryje magnet. Po těchto úpravách jsou figury připraveny pro manipulaci.



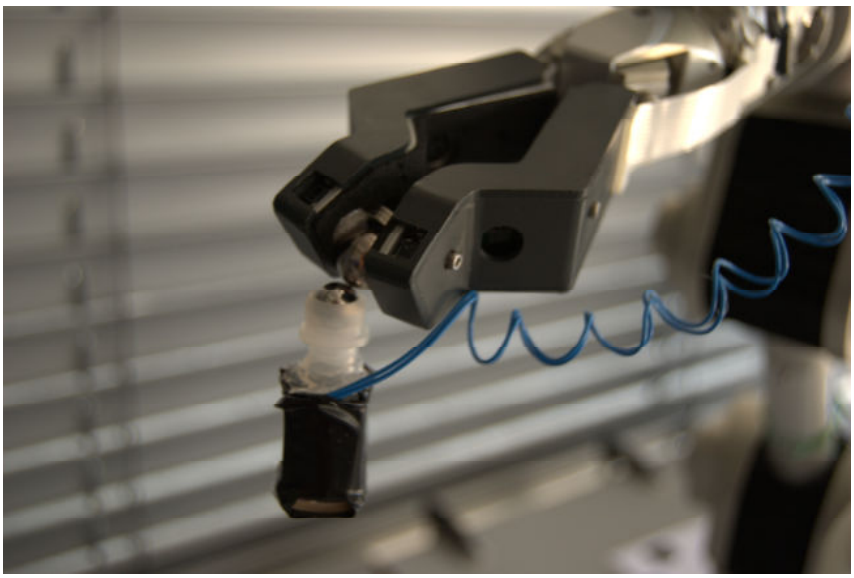
Obr. 26 Upravené figury zasazené do šachovnice.

## 9.13 Fyzická úprava uchopovače manipulátoru

Manipulátor Katana 300s, který je použit v této práci, je vybaven univerzálním uchopovačem. Protože by při manipulaci s figurami tímto uchopovačem mohla vznikat nepřesnost, je pro přesun figur použit elektromagnet. Jeho použití v této úloze má hned několik výhod. Největší z nich je vycentrování figury vůči souřadnému systému manipulátoru při přesunu. Pokud je koncový bod manipulátoru odchýlen od přesného středu figury a dojde k sepnutí elektromagnetu, figura je silou magnetu přitažena a odchylka se odstraní.

Díky zavěšení elektromagnetu na adaptéru ve tvaru koule nemusí program uvažovat úhel uchopovače. Silou gravitace je vždy zaručen kolmý směr závěsu. Jako kulový adaptér slouží další neodymový magnet s magnetickou silou tři kilogramy, což je pro tuto úlohu dostatečné. Magnet zapadne do výdutí v uchopovači a tím je zaručeno přesné umístění. Aby nedocházelo k protáčení kulového magnetu v uchopovači jsou na jedné straně umístěny další magnety, zabraňující tomuto pohybu. Aby nedocházelo k ovlivnění úhlu volně visícího elektromagnetu dráty, přivádějící napětí, jsou natočené do spirály.

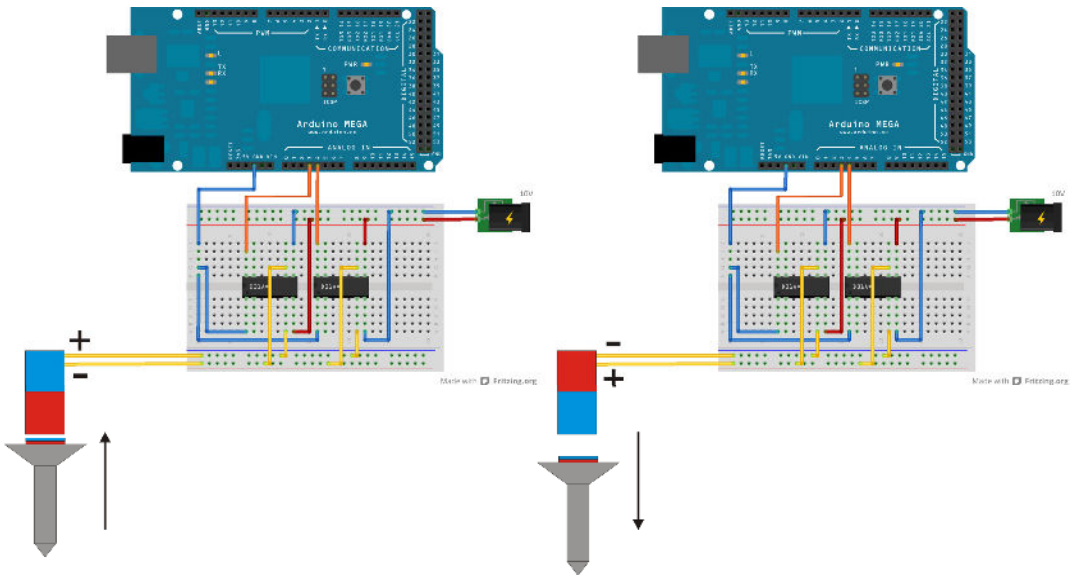
Jako elektromagnet je v této práci použito spínací relé. Elektromagnetická síla, běžně fungující k přitažení kotvy a sepnutí kontaktu, je dost velká aby mohla manipulovat s figurami. Síla magnetu, umístěného na figuře je natolik velká, že se udrží na adaptéru i bez připojeného elektrického napětí. Proto je nutné při pokládání otočit polaritu napájení relé a změnit tak pól jádra. Protože na sebe začnou působit dva magnety natočené stejným pólem, vzniká odpudivá síla, která odhodí figuru na její nové souřadnice.



Obr. 27 Upevnění elektromagnetu na kulovém adaptéru.

## 9.14 Ovládání elektromagnetu

Spínání elektromagnetu je řešeno přes vývojovou platformu Arduino. Komunikace s řídicím programem je realizována pomocí sériového portu. Na straně PC je deklarován název portu, v tomto konkrétním případě COM4, a nastavena znaková rychlost 9600 baudů. Po správném nastavení lze vývojové platformě posílat zakódované příkazy na spínání jednotlivých relé, ovládající polaritu napájení elektromagnetu. Celá komunikace je ošetřena odchycením výjimek. Pokud komunikace z nějakého důvodu nemůže proběhnout, uživatel je o této situaci informován. Po odstranění problému komunikace a fyzickém přesunu figury lze pokračovat dále ve hře.
Elektronický obvod, který se stará o spínání je založen na dvojici relé. Každé relé sepne elektromagnet s jinou polaritou napájení a tím je realizováno otočení pólu jádra, které je nutné pro správnou funkcionalitu přesunu figur. Ovládání jednotlivých relé je řešeno standardem TTL. Jako odezvu na zaslaný kód z programu nastaví vývojová platforma na požadovaný pin hodnotu *HIGH* nebo *LOW*. Nastavení komunikace je obdobné jako na straně PC. Ve vývojové platformě je poté spuštěna nekonečná smyčka pro detekci přicházející komunikace. Při rozeznání kódu je na jednotlivé piny nastavena požadovaná hodnota. Při spínání relé probíhá kontrola, která zabrání sepnutí obou najednou, aby nedošlo ke zkratování zdroje. Napájení elektromagnetu je možné realizovat libovolným stejnosměrným zdrojem s výstupním napětím 10 voltů a zatížitelností 500 miliampér.




Obr. 28 Schéma zapojení obvodu pro spínání elektromagnetu.

## 9.15 Tah manipulátoru

Po zjištění pozice všech figur na hrací desce a zvolení nejlepšího možného tahu následuje tah manipulátoru. Pro výpočet souřadnic manipulátoru jsou použity koeficienty převodu, jejichž výpočet je popsán výše v tomto textu. Pomocí vzorců (10) a (11) jsou tyto souřadnice získány. Souřadnice $z$ je dána výškou figur a šachovnice.

$$x_{manipulátor} = \left(x_{figura} / PxToMm_{koeficient}\right) + x_{koeficient} \tag{10}$$

$$y_{manipulátor} = \left(\left(1080 - y_{figura}\right) / PxToMm_{koeficient}\right) + y_{koeficient} \tag{11}$$

Pokud se jedná o tah jedné figury na prázdné pole, jsou manipulátoru zaslány souřadnice přesunované figury a výška, která je větší než výška figury. Souřadnice figury jsou zjištěny dynamicky ze snímaného obrazu, výška je zadána. Do této polohy se manipulátor přesune základní nastavenou rychlostí principem inverzní kinematiky. Po dosažení této polohy se manipulátor přepne na nižší rychlost a nad figuru se přesune lineárním pohybem. Tím je zaručeno, že při pohybu nebude narážet do ostatních figur. V této chvíli je také sepnut elektromagnet na opačný pól, než je na vrcholu figury. Po dostatečném přiblížení k figuře je tato elektromagnetem přitažena a dojde k automatickému vycentrování, díky vlastnostem magnetů. Souřadnice cílového pole jsou získány z virtuální šachovnice, vytvořené při startu programu. Přichycená figura je následně vyzvednuta lineárním pohybem do takové výšky, aby nebyly ostatní figury ohrožené pohybem na cílové pole. Stejným principem lineárního pohybu je figura umístěna na cílové pole. Po dosažení cílové pozice je elektromagnetu otočena polarita zdrojového napětí a dojde k přepólování magnetu a tím k odhození figury. Figury i šachovnicová deska jsou navrženy tak, aby při odhození figury došlo k vycentrování na pozici. Tímto opatřením je zvýšena spolehlivost celé úlohy.

V případě tahu na obsazené pole je nejdříve figura na cílovém poli odhozena mimo šachovnici. Postup uchycení figury je stejný, jako v předchozím případě. Cílové souřadnice pro odhození figury jsou mimo herní desku úlohy. Poté následuje tah figury na již volné cílové pole.

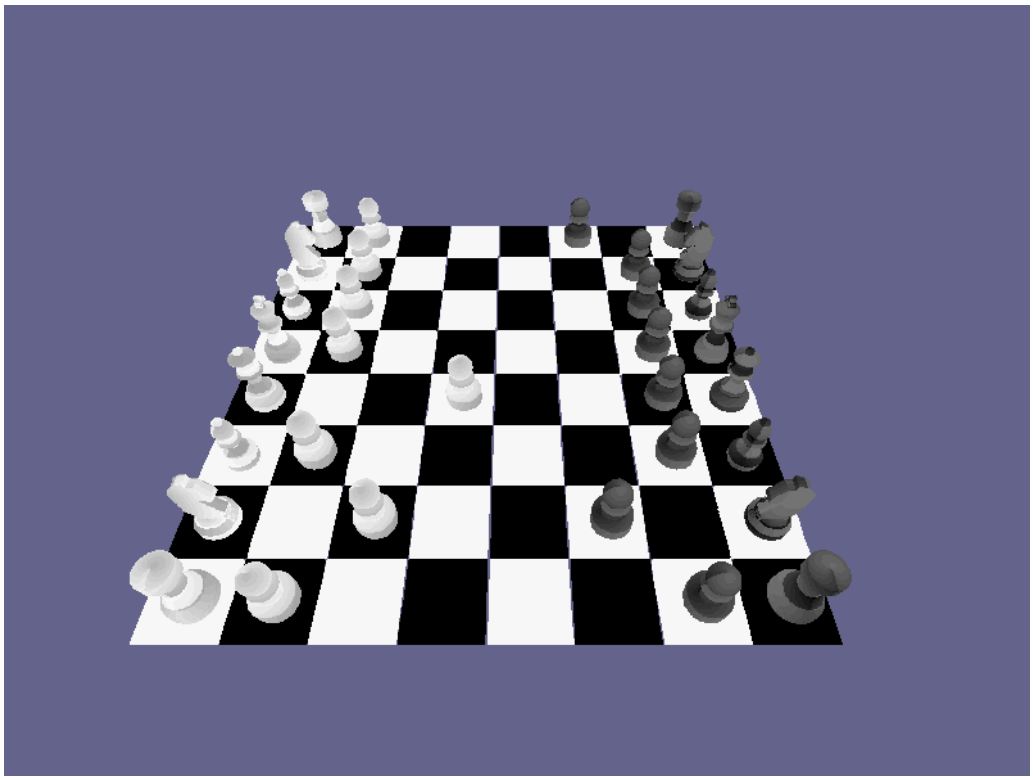
## 9.16 Vizualizace herní scény

Pro vizualizaci herní scény jsou použity 3D modely šachů a šachovnice. Scéna je generována pomocí 3D Engine Irrlicht. Vždy po tahu počítačového nebo lidského hráče je aktualizována pozice figur. Figury jsou rozmístěny na 3D scéně podle šachovnice, kterou si uchovává šachový algoritmus ve své vnitřní reprezentaci. Tato šachovnice je matice textových hodnot, označující jednotlivé figury a jejich pozice. Před průchodem tímto polem jsou nahrány 3D modely figur. Podle druhu je jim přiřazena černá nebo bílá textura. Při vlastním průchodu polem se upravují souřadnice, kam se má figura umístit a probíhá testování, která figura bude na kon-

krétní souřadnici stát. Aby zobrazování fungovalo korektně, musí být engine Irrlich spuštěn v druhém vláknu aplikace.

Renderování probíhá pomocí softwarového renderu, obsaženého v engine Irrlicht. Jeho výhodou je nezávislost na typu renderu instalovaného v PC. Protože je softwarové renderování poměrně náročné na výkon počítače a zvládne jen omezené množství objektů, je nutné do scény zasadit jen nezbytné množství renderovaných objektů.

3D zobrazení šachovnice je v programu realizováno zejména z prezentačních důvodů. Při hře může lidský hráč moci sledovat, jak se zobrazení mění v závislosti na snímání kamery a pohybu manipulátoru. Základní 3D modely šachových figur před integrací do systému vizualizace byly převzaty z práce studenta Josefa Truhláře do předmětu Počítačová Grafika I, vyučované na provozně ekonomické fakultě Mendelovy univerzity.



Obr. 29 3D scéna šachovnice

# 10 Ověření spolehlivosti řešení

Ověření spolehlivosti celého řešení je součástí cíle této práce. Následující tabulky zobrazují procentuální úspěšnost v jednotlivých disciplínách této úlohy. Testování je zaměřeno na dvě základní části. Zpracování obrazu a manipulace.

## 10.1 Spolehlivost zpracování obrazu

Chyby ve zpracování obrazu jsou rozděleny na tři skupiny. První z nich se zabývá prvotním předzpracováním snímku a nalezením správného počtu vzorů. Tyto chyby nejsou pro program kritické. Znamenají pouze větší časovou náročnost při snímání tahu lidského hráče. Druhou skupinou jsou chyby určení tahu lidského hráče, kdy je nesprávně tah označen jako neplatný. V tomto případě lze ve hře dále pokračovat, ale snižuje se uživatelská přívětivost. Lidský hráč musí použít příkaz pro potvrzení opravy tahu. Třetí skupinou jsou chyby kritické, po jejichž výskytu nelze ve hře dále pokračovat a partie je ztracena.

Tabulka 3 Procentuální úspěšnost při zpracování obrazu

| Počet snímání | Redundance snímků | Počet chybných identifikací tahů | Počet kritických chyb |
|---|---|---|---|
| 200 | 632 | 6 | 0 |
| 100% | 316% | 3% | 0% |

## 10.2 Spolehlivost manipulace

Manipulační spolehlivost je rozdělena do dvou skupin. Chyby manipulace figur, které vyžadují zásah lidského hráče, ale lze pokračovat ve hře. Může se vyskytnout chyba v ovládání elektromagnetu, nebo chyba v umístění figury do cílového pole. Dále kritické chyby manipulátoru, po jejichž výskytu nelze dále pokračovat ve hře. Zde se může projevit chyba hardware nebo chyba výpočtu cílových souřadnic, které způsobí kolizi manipulátoru.

Tabulka 4 Procentuální úspěšnost při manipulaci s figurami

| Počet manipulací | Vyžadován zásah | Kritické chyby manipulátoru |
|---|---|---|
| 200 | 42 | 1 |
| 100% | 21% | 0,5% |

# 11 Diskuze

Při konstrukci šachového automatu se u každého modulu projevily problémy, se kterými nebylo při návrhu počítáno. Každá taková nesnáze se negativně projevila na celkové časové náročnosti celkového řešení. Z tohoto důvodu je zde velký prostor pro rozšíření celé úlohy. Návrh programu je koncipován tak, aby bylo možné nahradit každý modul za jiný.
Při hodnocení spolehlivosti řešení bylo ve 42 případech z 200 manipulací nutné zasáhnout a manuálně upravit tah počítačového hráče. Tato spolehlivost bude zvýšena přesnějším uchycením elektromagnetu a větším zatížení spodní části figur. Nebude tak docházet k rozkmitání figury při manipulaci a bude přesněji umístěna. Ke zvýšení manipulační spolehlivosti také přispěje chystaný vylepšený systém kalibrace.

Hlavním možným rozšířením je detekce standardních šachových figur. Toto rozpoznávání je možné realizovat dvěma základními způsoby. Jeden způsob je založen na rozpoznávání analýzou barevného prostoru. Tento způsob je například použit v projektu MarineBlue, popsaný v části analýza současného stavu řešené problematiky. Tato detekce ovšem vyžaduje barevně odlišené figury. S klasickými černými a bílými figurami na černobílé šachovnici nelze pracovat. S tímto přístupem by k řešení nebylo potřeba figury opatřovat vzory pro rozpoznání v obraze. Již toto rozšíření by přineslo zlepšení celkového vzezření úlohy. Druhou možností je využít kvalitnější kamerové vybavení. Při použití podobné kamery, která je využita u robotu Gambit z kapitoly analýza současného stavu řešené problematiky je možné detekovat i více typových sestav šachových figur. Tohoto rozpoznání lze také dosáhnout použitím více kamer, snímajících herní prostor.

Rozšíření o rozpoznávání standardních šachových figur sebou také přinese otevření problematiky manipulace s těmito figurami. Řešení pomocí elektromagnetu má své výhody i nevýhody. V případě, že by standardní figury byly vybaveny nenápadným magnetem malé velikosti na horní straně, bylo by možné využít vycentrování při uchopení. Tím by došlo ke smazání případné nepřesnosti. Při použití elektromagnetu pro manipulaci by bylo dobré zvážit možnost miniaturizace jeho spínání a napájení. Vývojovou platformu Arduino by bylo možné nahradit USB relé. Jako zdroj by mohl sloužit malý adaptér nebo přímo elektronika v horní části manipulátoru.
Pokud by bylo zvažováno použití univerzálního uchopovače, kterým je manipulátor Katana 300s vybaven, musel by být zřejmě hardwarově upraven. Jedna z jeho výhod je osazení vzdálenostních senzorů a senzorů tlaku v čelistech. Těmito senzory by bylo možné vyrovnávat nepřesnosti získané ze zpracování obrazu z kamer. Zásadní nevýhodou je různá výška a tvar jednotlivých figur. Pravděpodobnost špatného uchycení či odhození figury je poměrně vysoká. V úvahu také připadá kombinované rozšíření. Použití univerzálního uchopovače a zároveň upravit spodní základnu figur. Figury by neztratily klasický vzhled, ale dolní část by byla normalizována pro uchopení.

Část 3D vizualizace by také mohla být rozšířena. Renderování je v tomto řešení prováděno staticky. Pro nové rozložení se vygeneruje rozestavění figur a toto je zobrazeno. Protože je toto řešení vypracováno jako prezentační, jistě by bylo zajímavé, kdyby se přidala animace pohybu. Pro vyhození figury protihráče by mohla být zakomponována zajímavá animace. Dále by bylo možné pomocí grafického uživatelského rozhraní ovládat pohyby figur. Hráč by mohl táhnout pomocí počítače a manipulátor by vykonával fyzickou práci. To vše by umožňoval grafický engine Irrlicht.

I modul šachového algoritmu by se mohl zefektivnit. Jednou z možností je vylepšení stávajícího šachového algoritmu Huo Chess. Další možností je návrh a implementace vlastního šachového algoritmu. Avšak při široké nabídce volně dostupných kvalitních šachových algoritmů by to nebylo efektivní využití časových zdrojů.

Výše jsou popsány návrhy na poměrně zásadní změny v řešení. Jedním z menších rozšíření by mohlo být implementování hlasového ovládání a komunikace s lidským hráčem. V představeném řešení také není implementována funkce vrácení figury na původní místo při neplatném tahu lidského hráče. Pro kompletnost programu a zábavnost při hraní jistě užitečná funkce. Odhadovaná časová náročnost těchto vylepšení není příliš vysoká.

Pro detekování šachovnice při každém tahu hráče, by musela být kamera umístěna výše nad hrací deskou. Také by byl nutný vyšší výkon počítače. Současné řešení umožňuje umístit šachovnici s figurami do libovolné pozice ve snímaném obrazu. Po inicializaci hry se však již nesmí odchýlit od původní pozice.

# 12 Závěr

Cílem diplomové práce bylo navrhnout a implementovat šachový automat. Jednou z možností využití tohoto automatu je propagace Mendelovy univerzity v Brně. Dále vytvoření vzorového projektu pro výuku v předmětech vyučovaných na provozně ekonomické fakultě. Návrh vycházel ze stanovených požadavků na dané řešení. Vyčerpávající přehled požadavků je uveden výše v této práci.

Cíl práce byl splněn a detailní popis řešení je v této práci uveden. Během návrhu i konstrukce bylo nutné se vyrovnat s velkým množstvím obtíží u jednotlivých modulů. Nejvíce problémů se projevilo při integraci jednotlivých modulů do celkového řešení. Po zvládnutí těchto problémů a uvedení celého systému do funkčního stavu byla ověřena jeho spolehlivost a přehledně znázorněna v kapitole ověření spolehlivosti řešení. Fotodokumentace šachového automatu při prezentaci univerzity je uvedena v příloze.

# 13 Literatura

Arduino - HomePage. Arduino [online]. 2006 [cit. 2013-03-19]. Dostupné z: http://arduino.cc/

BRADSKI, G. R. -- KAEHLER, A. Learning OpenCV : computer vision with the OpenCV library. Farnham: O'Reilly, 2008. 555 s. ISBN 978-0-596-51613-0.

Camera calibration With OpenCV. OpenCV [online]. 2011 [cit. 2013-03-18]. Dostupné z: http://docs.opencv.org/doc/tutorials/calib3d/camera_calibration/camera_calibration.html

GNU Chess. GNU Operating System [online]. 1996 [cit. 2013-04-30]. Dostupné z: http://www.gnu.org/software/chess/

GONCALVES, José, José LIMA a Paulo LEITAO. DEPARTAMENTO DE ELECTROTECNIA, Quinta de Sta Apol´onia. CHESS ROBOT SYSTEM: A MULTI-DISCIPLINARY EXPERIENCE IN AUTOMATION. 2013, 8 s.

Huo Chess - Home. Huo Chess [online]. 2006 [cit. 2013-03-18]. Dostupné z: http://huochess.codeplex.com/

Chess Playing Robot. Ebrahim Jahandar official website [online]. 2013 [cit. 2013-04-18]. Dostupné z: http://www.jahandar.ir/Chess-Playing-Robot/

Chess-king. Chess King [online]. 2013 [cit. 2013-04-18]. Dostupné z: http://chess-king.com/konstantin-kosteniuks-chesska-is-2012-world-robot-champion/#more-818

Irrlicht Engine. Irrlicht [online]. 2013 [cit. 2013-03-18]. Dostupné z: http://irrlicht.sourceforge.net/

Limcorp. LimCorp.net [online]. 2013 [cit. 2013-04-29]. Dostupné z: http://limcorp.net/2010/microsoft-lifecam-high-hd-sensor

MATUSZEK, Cynthia, Brian MAYTON, Roberto AIMI, Marc PETER DEISENROTH, Liefeng BO, Robert CHU, Mike KUNG, Louis LEGRAND, Joshua R. SMITH a Dieter FOX. Gambit: A Robust Chess-Playing Robotic System. 2013, 7 s.

MELSOFT-Software. MITSUBISHI ELECTRIC. Mitsubishi automation [online]. 2013 [cit. 2013-03-21]. Dostupné z: http://www.mitsubishi-automation-cz.com/products/software_content.html

MITSUBISHI ELECTRIC. MELFA robots. 2010. Dostupné z: http://www.consys.ru/sites/default/files/documents/robots_0.pdf

NEURONICS AG. Katana Native Interface dokumentace: KNI. 2006, 14 s.

NEURONICS AG. Katana 450: Katana Native Inteface API Documentation. 2008, 347 s.

NEURONICS AG. Návod k použití robotů Katana. 2006, 62 s.

Neuronics Katana. Katana exactech [online]. 2011 [cit. 2013-03-18]. Dostupné z: http://katana.exactec.com/

Ogre. TORUS KNOT SOFTWARE LTD. Object-Oriented Graphics Rendering Engine [online]. 2000 [cit. 2013-04-29]. Dostupné z: http://www.ogre3d.org/

OpenCV. Open Source Computer Vision Library [online]. 2013 [cit. 2013-03-18]. Dostupné z: http://opencv.org/

ROUŠ, Robert. Řízení robotu KATANA v prostředí Control Web. Brno, 2012. Diplomová práce. Mendelova Univerzita v Brně.

Stockfish. Stockfish Chess [online]. 2010 [cit. 2013-04-30]. Dostupné z: http://stockfishchess.org/

TOMÁŠEK, Eduard. Systém ovládání robota Katana prostřednictvím programovacího jazyka C. Brno, 2010. Bakalářská práce. Mendelova univerzita v Brně.

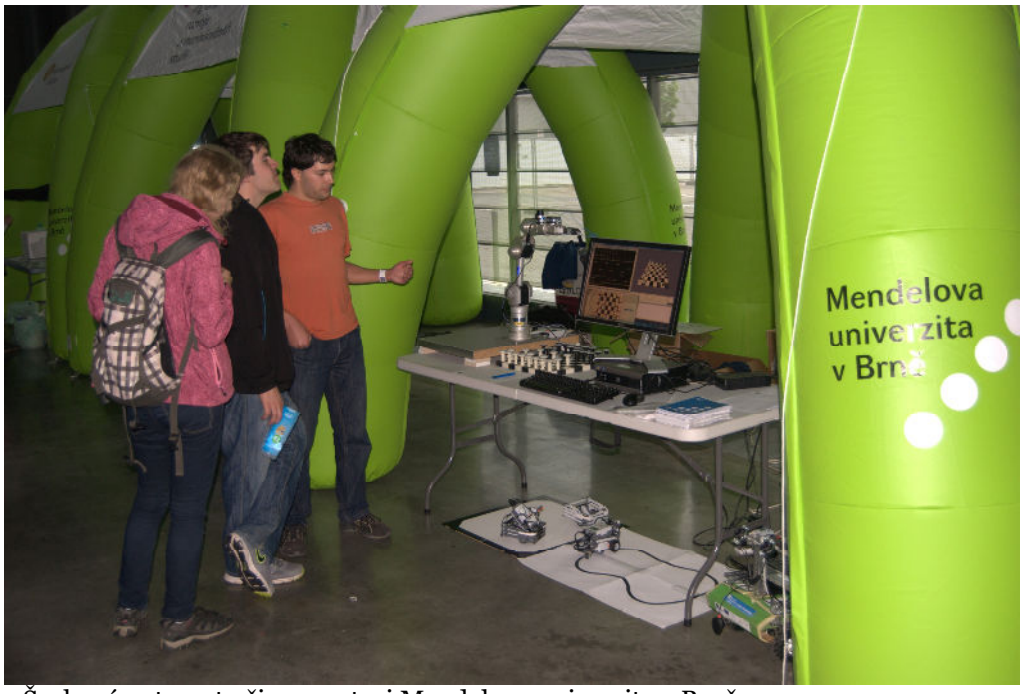
URTING, David a Yolande BERBERS. DEPARTMENT OF COMPUTER SCIENCE CELESTIJNENLAAN. MarineBlue: A Low-Cost Chess Robot. 2013, 6 s.

VisionLab. Mitov Software [online]. 2013 [cit. 2013-04-29]. Dostupné z: http://www.mitov.com/products/visionlab#overview

What was the Turk?. CONJECTURE CORPORATION. WiseGEEK [online]. 2003 [cit. 2013-04-17]. Dostupné z: http://www.wisegeek.com/what-was-the-turk.htm

# Přílohy

# A Šachový automat při prezentaci




Obr. 30 Šachový automat při prezentaci Mendelovy univerzity v Brně


Obr. 31 Šachový automat při prezentaci Mendelovy univerzity v Brně

# B Arduino zdrojový kód

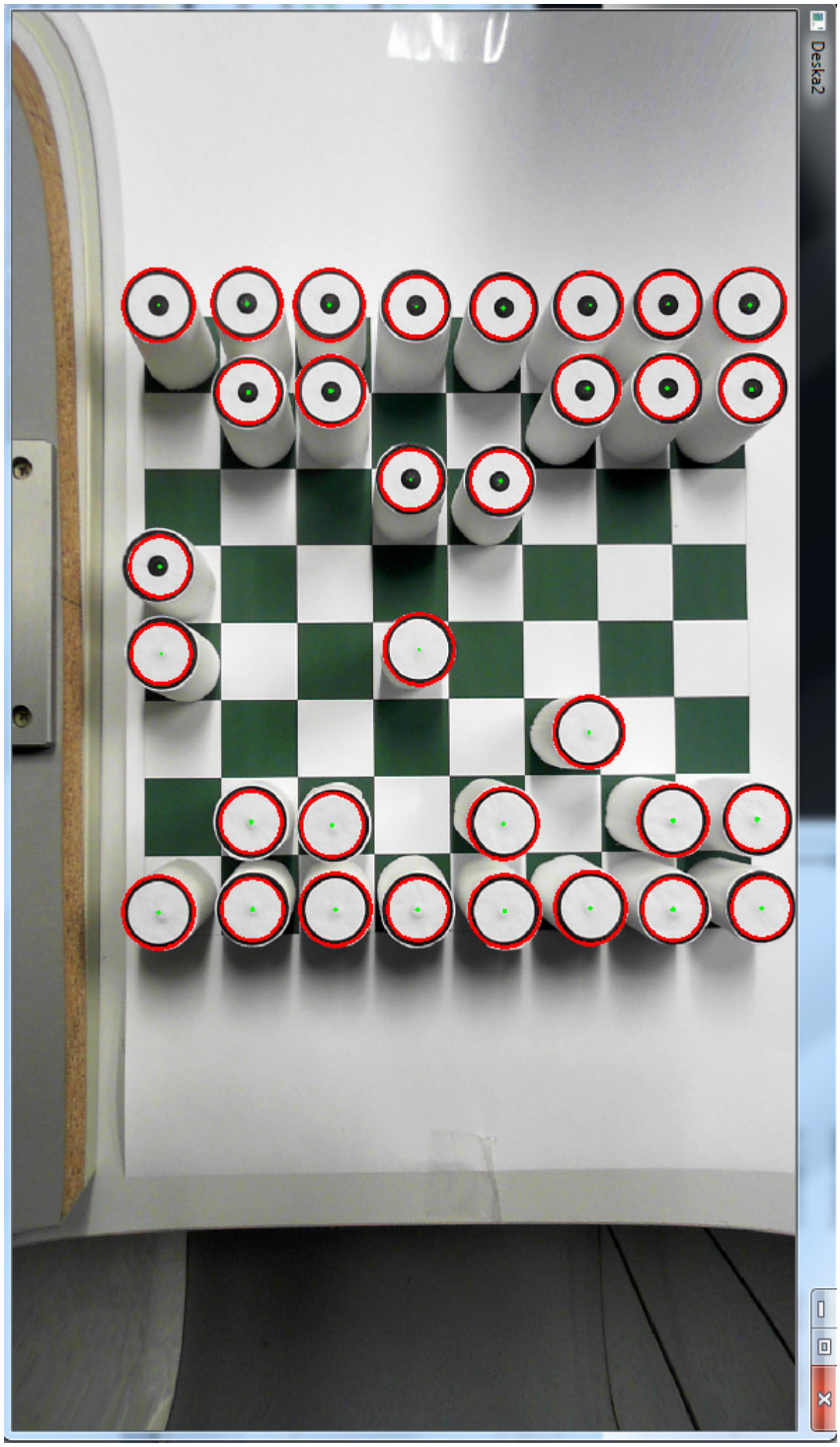
```
int plusPin = 4;
int minusPin = 5;
int state=0;

void setup() {

   pinMode(plusPin, OUTPUT);          // nastavení výstupních pinů
   pinMode(minusPin, OUTPUT);
   Serial.begin(9600);                //musí být nastavené stejně na straně PC

}

void loop() {
  if (Serial.available() > 0)
  {
   state = Serial.read();             // Použito pro čtení příchozích dat

   switch(state)                      //Testování příchozí hodnoty
   {
    case '1':                         // Sepnutí elektromagnetu pro přítah
      digitalWrite(minusPin,LOW);
      digitalWrite(plusPin,HIGH);
    break;
    case '0':                         // Vypnutí napájení relé 1
      digitalWrite(plusPin,LOW);
    break;
    case '2':                         // Sepnutí elektromagnetu pro odhození
      digitalWrite(plusPin,LOW);
      digitalWrite(minusPin,HIGH);
    break;
    case '3':                          // Vypnutí napájení relé 2
      digitalWrite(minusPin,LOW);
    break;
    default:
    break;
   }
  }
}
```

# C Ukázka podobných řešení

Obr. 32 Šachový robotický multidisciplinární systém
Ztdroj: Převzato od Goncalvese (2013)

Obr. 33 Šachový robot gambit při hře.
Zdroj: Převzato od Matuzseka (2013).

# D Prototyp figur pro vývoj




Obr. 34 Prototyp figur pro vývoj software před 3D tiskem.