Interaktivní hra Piškvorky Člověk-Robot: Integrace Počítačového Vidění a Robotické Ruky

Michal Průšek
ČVUT–FJFI
prusemic@cvut.cz

26. května 2025

1 Úvod

Tento projekt se zabývá výzvou vytvoření interaktivního herního systému člověk-robot, který plynule integruje počítačové vidění, umělou inteligenci a robotické řízení. Vyvinutá aplikace Piškvorky (Tic-Tac-Toe) umožňuje lidem hrát proti AI protivníkovi na fyzické herní desce, přičemž tahy AI provádí robotická ruka uArm Swift Pro [6].

Hlavním cílem je demonstrovat efektivní interakci člověka s robotem prostřednictvím hry, kombinující vnímání v reálném čase s fyzickou manipulací. Systém musí přesně detekovat stav hry, vypočítat optimální tahy a fyzicky kreslit symboly na desku, to vše při zachování poutavého uživatelského zážitku. Aplikace je navržena pro robustní fungování, zahrnující detekci herní mřížky a symbolů, kalibraci ruky a adaptivní AI.

2 Vstupní data

Aplikace zpracovává několik datových toků pro udržení povědomí o stavu hry:

2.1 Vizuální vstup

Primárním vstupním zdrojem je webkamera snímající fyzickou herní desku. Video stream je zpracováván v reálném čase pro detekci:

- Struktury mřížky: Detekce 16 klíčových bodů (pro herní mřížku 3x3 buněk, klíčové body jsou umístěny v rozích a průsečících desky) pomocí modelu YOLOv8 Pose [7].
- Herních symbolů: Symboly X a O umístěné na desce hráčem nebo robotickou rukou, detekované pomocí modelu YOLOv8 [7].
- Orientace a perspektivy desky pro přesné mapování buněk pomocí homografie.

Kamera je typicky specifikována indexem (výchozí: 0) pomocí parametru příkazové řádky. Systém preferuje externě připojené kamery.

Datasety obrázků byly anotovány v platformě CVAT [1] a jsou dostupné zde TicTacToe-KAGGLE. Případně pak na vyžádání.

2.2 Uživatelská interakce

Uživatelé mohou interagovat se systémem prostřednictvím:

- Fyzického umístění symbolů na herní desku.
- Grafického uživatelského rozhraní (GUI) postaveného na PyQt5 [5], které zobrazuje stav hry, notifikace a umožňuje sledování statistik.
- Konfiguračních parametrů zadávaných přes příkazovou řádku při spuštění aplikace (např. `python -m app.main.main_pyqt` nebo `run.py`):
  - `--camera <index>`: Index kamery (výchozí: 0).
  - `--debug`: Zapnutí debugovacího režimu.
  - `--difficulty <0-10>`: Obtížnost AI (výchozí: 5).
  - `--max-fps <fps>`: Maximální FPS pro detekční vlákno.

2.3 Kalibrační data

Robotická ruka vyžaduje data pro kalibraci oko-ruka uložená v souboru `app/calibration/hand_eye_calibration.json`. Tato data mapují souřadnice kamery (pixely) na souřadnice pracovního prostoru robota (fyzické jednotky) pomocí 3x3 perspektivní transformační matice. Soubor také obsahuje kalibrované výškové úrovně. Kalibrace se provádí spuštěním skriptu `python -m app.calibration.calibration` a zahrnuje detekci mřížky, korekci homografie pomocí RANSAC [2], sběr korespondenčních bodů (UV z kamery $\leftrightarrow$ XY robota) a výpočet matice metodou nejmenších čtverců.

3 Metody a algoritmy

3.1 Implementace AI strategie

AI protivník využívá algoritmus Minimax s alfa-beta prořezáváním pro výběr optimálního tahu [3]. Strategie je implementována pomocí vzoru Strategy a zahrnuje:

- **BernoulliStrategySelector**: Mapuje uživatelsky zvolenou obtížnost (0-10) na pravděpodobnost (0.0-1.0). Na základě této pravděpodobnosti volí (z Bernoulliho rozdělení) mezi:
  - Náhodnou strategií (pro nižší obtížnosti).
  - Minimax strategií (pro vyšší obtížnosti).

  Vyšší obtížnost znamená častější použití optimálních tahů z Minimaxu.
- **Minimax algoritmus**:
  - Alfa-beta prořezávání pro efektivitu.
  - Heuristiky pro běžné scénáře (obsazení středu, rohů).
  - Prioritizace rychlých výher a odkládání proher.
  - Hodnocení pozic: +10 pro výhru AI, -10 pro prohru AI, 0 pro remízu.

Správa herního stavu je centralizovaná v `GameController.authoritative_board`. Detekce tahu (zda hraje robot) se určuje podle počtu symbolů na desce (lichý počet = tah robota).

3.2 Řídicí systém robotické ruky

Ovládání robotické ruky uArm Swift Pro [6] je hierarchické a využívá knihovnu uArm-Python-SDK [4]:

1. **ArmMovementController**: Zajišťuje vysokoúrovňové pohyby (např. kreslení symbolu X nebo O).
2. **ArmThread**: Zpracovává příkazy pro robotickou ruku asynchronně pomocí fronty příkazů, což zajišťuje neblokující API.
3. **ArmController**: Poskytuje nízkoúrovňové ovládání specifické pro uArm Swift Pro.

4 Závěr

Tento projekt úspěšně demonstruje integraci počítačového vidění, umělé inteligence a robotického řízení v interaktivním herním kontextu. Aplikace Piškvorky s robotickou rukou uArm Swift Pro [6] slouží jako důkaz konceptu pro spolupráci člověka s robotem, představující schopnosti vnímání v reálném čase a fyzické manipulace.

Aplikace je postavena na robustní architektuře zahrnující dvoustupňovou detekci pomocí YOLO modelů [7], adaptivní AI s Minimax algoritmem [3] a přesné ovládání robotické ruky kalibrované pomocí perspektivní transformace. Komplexní systém konfigurace, detailní kalibrační proces a vícevláknové zpracování zajišťují flexibilitu, přesnost a plynulost.

Videa chodu aplikace a fotka aparatury jsou k dispozici zde: Google Drive V případě dalšího vývoje je nejaktuálnější verze k dispozici na mém osobním Github Github-michalprusek-TicTacToe

Reference

[1] CVAT team and contributors. Computer Vision Annotation Tool (CVAT). `https://github.com/opencv/cvat`, 2018–2024.

[2] Martin A. Fischler and Robert C. Bolles. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Communications of the ACM*, 24(6):381–395, June 1981.

[3] Sumit Shevtekar, Mugdha Malpe, and Mohammed Bhaila. Analysis of game tree search algorithms using minimax algorithm and alpha-beta pruning. *International Journal of Scientific Research in Computer Science, Engineering and Information Technology*, pages 328–333, 11 2022.

[4] UFACTORY Developer Team. uArm-Python-SDK - Python Library for uArm. `https://github.com/uArm-Developer/uArm-Python-SDK`, 2023. Přístup 21. září 2024.

[5] The Qt Company Ltd and Riverbank Computing Limited. Pyqt5, 2023.

[6] UFACTORY. *uArm Swift Pro User Manual*, 2017.

[7] Ultralytics. YOLOv8. `https://github.com/ultralytics/ultralytics`, 2023. Přístup 21. září 2024.