Monash University

Requirements Specification

Monash University Clayton Campus
Electrical and Computer Systems Engineering
Final Year Project

Chess Playing Robot

Supervisor: Dr Lindsay Kleeman

Contributors: Eric Horng
Matthew King

Date: 27 / 03 / 2019

1. TABLE OF CONTENTS

2. INTRODUCTION

2.1 Objectives

The purpose of this document is to specify the core and optional requirements for the Chess Playing Robot developed by Eric Horng and Matthew King for their Final Year Project in 2019.

2.2 Context

This project involves the design and construction of a 3D printed robotic arm that can play chess against a human opponent. The robot can analyse the board, move chess pieces and remove them when they have been eliminated. A specialty gripper will be developed and will be used to pick up the cylindrical chess pieces and a webcam will be used to detect the position of each piece.

The device will consist of three modules - the PSOC which will handle movement of the arm, the computer which will handle image processing and chess playing, and finally the webcam, which will take pictures of the board for the computer to process (See Figure 1). The camera and PSOC will be connected to the computer with a USB cable. A UART will be used to communicate with the PSOC and send serial data.

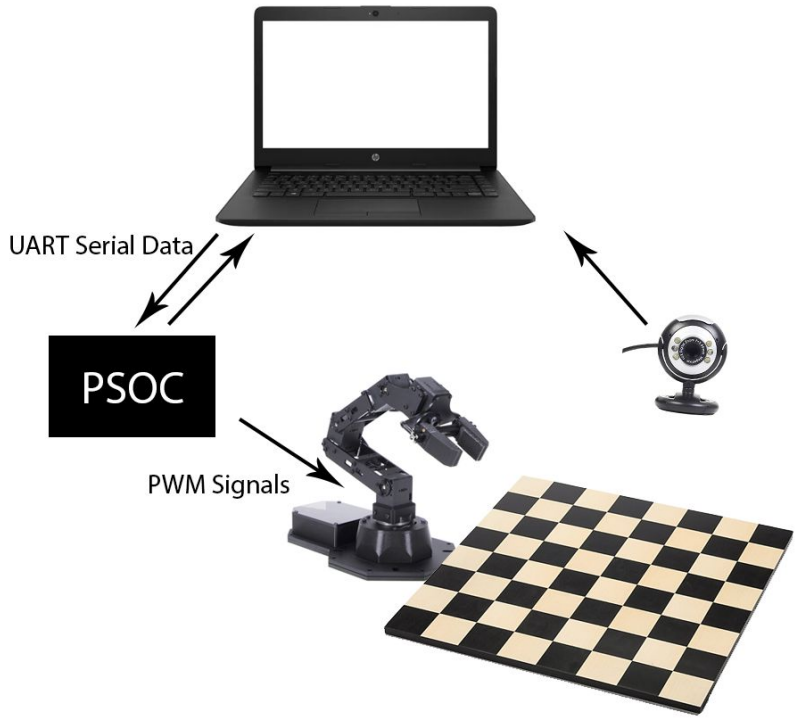

Figure 1. Diagram showing the interaction between arm, computer and webcam modules

Within the computer, two separate functions will handle computer vision and chess AI operation. The computer vision program will take a snapshot after each turn, apply edge/feature to all the pieces to identify them and save it to a 64x64 grid which will be handed to the chess AI program. The chess AI program will analyze the board, decide the next best move and send two coordinates- dictating the piece to move and where to move it- to the robotic arm which will perform the action.

3. REQUIREMENTS

3.1 High Level Requirements

Code	Type	Requirement
C.1	Core	The computer should be able to locate and identify individual chess pieces.
C.2	Core	The robotic arm should be able to pick up chess pieces and place them down.
C.3	Core	The computer should be able to determine the next optimal move given the state of the chess board.
O.1	Optional	The system should be able to reset the board for a new game.

3.2 Financial Requirements

Code	Type	Requirement
C.4	Core	The cost of the project should not exceed $200 AUD.

3.3 Physical Requirements

3.3.1. Physical Characteristics Requirements

Code	Type	Requirement
C.5	Core	The arm should not weigh more than 10kg.
C.6	Core	The arm should be constructed out of a hard, durable material (3D printed plastic, acrylic, plywood, sheet metal).
C.7	Core	The arm should not break or crack during normal operating conditions.
C.8	Core	The arm should be long enough to reach all squares on the chess board.
C.9	Core	The arm should be weighted appropriately so does not fall over during operation.

3.3.2 Circuit Board and Power Supply Requirements

Code	Type	Requirement
C.10	Core	The robotic arm should interface with the computer with a PSOC microprocessor.
C.11	Core	The robotic arm and modules should be powered with a 12V DC power pack.
O.2	Optional	The servos should be provided with enough power to not require separate power circuits

3.4 Functionality Requirements

3.4.1 Arm Functionality Requirements

Code	Type	Requirement
C.13	Core	The arm should be able to move a piece from one square to another square within a 30 seconds.
C.14	Core	The arm should be be able to place pieces within a square of a full-sized chessboard, with the piece not more than half out of the square.
C.15	Core	The robotic arm should be able to lift at least 50g.
O.3	Optional	The arm should be able to move pieces without knocking over other pieces.
O.4	Optional	The arm should be able to move eliminated pieces to a predefined zone.
O.5	Optional	The arm should be able to retrieve eliminated pieces if required in the game.

3.4.2 Computer Vision Functionality Requirements

Code	Type	Requirement
C.16	Core	The program should be able to differentiate between player pieces.
C.17	Core	The program should be able to differentiate piece types (pawns, queens, rooks etc.) with a recognition rate of at least 70%.
C.18	Core	The program should be able to determine the current state of the board within 30 seconds.
O.6	Optional	The program should be able to detect invalid player moves

O.7	Optional	The program should automatically detect if the board state has changed.
O.8	Optional	The program should be able to function in typical room lighting conditions.
O.9	Optional	The program should still function when used with a different full-sized chessboard.

3.4.3 Chess AI Functionality Requirements

Code	Type	Requirement
C.19	Core	The program should display the current state of the board when prompted (on the computer).
C.20	Core	The program should determine the next move within 20 seconds.
C.21	Core	The program should be able to recognise when the game is finished.
O.10	Optional	The program should automatically switch turns (dependent on computer vision) .
O.11	Optional	Different difficulties should be included for different user abilities.
O.12	Optional	The program should keep a history of all previous moves.

3.4.4 User Interface Functionality Requirements

Code	Type	Requirement
C.22	Core	The device should inform the user if either party has won or drawn (checkmate or stalemate).
C.23	Core	The device should have an indicator to show whose turn it is (LED, audio etc.).
C.24	Core	The device should allow you to fix erroneously placed pieces.
O.13	Optional	The device should have some input to allow the human player to forfeit and end the game prematurely.

3.5 Deadline Requirements

Code	Type	Requirement
C.25	Core	The robotic arm should be completed by the start of Semester 2.
C.26	Core	The computer vision component should be completed before Semester 2 Spark Night (Date T.B.A).
C.27	Core	The Chess A.I should be completed before Semester 2 Spark Night (Date T.B.A).
O.14	Optional	The User Interface should be completed before Semester 2 Spark Night (Date T.B.A).

3.6 Additional Requirements

Code	Type	Requirement
C.28	Core	A literature review with proper citations should be included with the final report (Date T.B.A).

3.7 Assumptions

Code	Type	Requirement
A.1	Assumption	Servos should have enough resolution to move within a square.
A.2	Assumption	The webcam has a high enough resolution to differentiate pieces.

3.8 Safety Requirements

Code	Type	Requirement
C.29	Core	There should be a readily accessible power switch to turn off robot in case of emergency.
O.15	Optional	Power circuits should be enclosed in a box to avoid short circuit and electrocution.
O.16	Optional	The arm should not move fast enough to hurt anyone caught within its range.

4. USE CASE SCENARIOS

4.1 Piece Identification and Position Detection

Condition	Description
Pre-conditions	Board is in a valid state, game is ongoing.
Trigger	When a piece is moved, the game has started or at the end of the player's turn
Action	The computer vision program takes a snapshot of the board using the webcam, identifies all the pieces on the board and their location and saves it. The current state is then sent to the chess AI for processing.
Post-conditions	No post-conditions.

4.2 Determining Next Move

Condition	Description
Pre-conditions	The human player's turn has just ended, move was valid and the board is in a valid state.
Trigger	When the current turn moves from the player to the computer and the computer vision program has sent the current state of the board to the chess AI.
Action	The computer will take the current state of the board and determine its next move.
Post-conditions	No post-conditions.

4.3 Checkmate or Stalemate Detection

Condition	Description
Pre-conditions	The board is in a state where there is a checkmate or stalemate.
Trigger	The chess AI program has analyzed the state of the board and determined the game is over.
Action	The computer will indicate that one player has won using an LED, display or sound.
Post-conditions	The game will end, and the computer will prompt the user to reset the board.

4.4 Moving Chess Piece

Condition	Description
Pre-conditions	It's the computer's turn.
Trigger	The chess AI has given the coordinates of a chess piece and an empty coordinate on the chessboard.
Action	The arm will pick up the piece, move the piece to the empty square and place it within the square.
Post-conditions	The computer's turn is over.

4.5 Eliminate Chess Piece

Condition	Description
Pre-conditions	It's the computer's turn.
Trigger	The chess AI has given the coordinates of a chess piece and an enemy occupied coordinate on the chessboard.
Action	The arm will pick up the piece and place it in a predetermined position outside the chessboard. The arm will then move the piece into the empty place.
Post-conditions	One piece has been removed from the game. The computer's turn is over.