장기 게임 개발 기획서

1. 개요

1.1 프로젝트 명칭

Korean Chess (Janggi) Game Engine & Platform

1.2 프로젝트 목표

전통 한국 장기를 디지털화하여 웹 기반 게임 플랫폼으로 구현하고, 규칙 엔진, UI/UX, AI 대국 기능을 포함한 완성도 있는 게임 환경 제공

1.3 핵심 요구사항

- 정확한 장기 규칙 구현 및 검증
- 실시간 멀티플레이 또는 AI 대국 지원
- 직관적인 게임 인터페이스
- 대국 기록 및 분석 기능
- 확장 가능한 아키텍처 (AI 학습, 고급 기능)

---

2. 게임 보드 및 초기 설정

2.1 보드 구성

속성	명세
크기	9 × 10 (교차점 기준)
총 교차점	90개
특수 영역	궁성 (Palace): 3 × 3, 대각선 4개 포함
좌표계	(x: 0-8, y: 0-9) 또는 (a-i, 0-9)

Table 1: 보드 기본 사양

2.2 궁성 (Palace)

- 각 진영(적/청)의 백행 끝에 위치 (3 × 3 영역)
- 적진(한/紅): y=0-2, x=3-5
- 청진(초/靑): y=7-9, x=3-5
- 중앙 교차점과 4개 모서리 교차점을 연결하는 대각선 4개 포함
- 궁(왕)과 사(보좌관)만 궁성 내에서 활동 가능

2.3 초기 기물 배치

기물	수량	점수	총점
궁 (King/General)	1	∞	-
사 (Advisor)	2	0	0
상 (Elephant)	2	3	6
마 (Horse)	2	5	10
차 (Chariot)	2	13	26
포 (Cannon)	2	7	14
졸 (Soldier)	5	2	10
합계	16	-	72

Table 2: 초기 기물 구성 및 점수

초기 배치 규칙[1]:

- 적진(한/홍): y=0-2 영역 (후수자가 먼저 배치)
- 청진(초/초): y=7-9 영역 (선수자가 나중에 배치)
- 마(馬)와 상(象)의 위치는 게임 시작 시 좌우 대칭 가능 (좌우 위치 교환 가능)
- 초(청)가 먼저 움직임
- 게임 시작 시 초는 72점, 한은 73.5점으로 계산 (후수 덤)

---

3. 기물 및 행마법 (Piece Movement Rules)

3.1 궁 (King/General) - 무한 점수

이동 범위:

- 궁성 내부의 직선 및 대각선을 따라 한 칸씩 이동
- 9개 위치 중 인접한 위치로만 이동 가능
- 중앙 위치에서 시작하여 대각선 및 직선으로 이동 가능
- 궁성을 절대 벗어날 수 없음

제약 사항:

- 상대 궁과 마주보는 상황 (비장 - bikjang) 발생 시 무승부
- 체크(장군) 상태에서 반드시 벗어나야 함
- 체크메이트 시 게임 패배

3.2 사 (Advisor/Guard) - 0점

이동 범위:

- 궁성 내부에서만 활동
- 직선 또는 대각선으로 한 칸씩 이동
- 궁과 함께 방어 기능

제약 사항:

- 궁성을 벗어날 수 없음
- 중요 방어 기물

3.3 상 (Elephant) - 3점

이동 방식[1]:

1. 먼저 직선 방향(상하좌우)으로 한 칸 이동
2. 그 다음 대각선으로 두 칸 이동
3. 순 이동 거리: 정방향으로 1칸, 사선으로 2칸 (L자 형태의 변형)

제약 사항:

- 경로상 중간에 다른 기물이 있으면 이동 불가 (점프 불가)
- 초기 위치의 직선 방향이 막혀있으면 그 방향으로 진출 불가
- 뒤로 이동할 때는 직선 방향 제약이 없음

이동 가능 위치: 2-4개 (현재 위치에 따라 가변)

3.4 마 (Horse) - 5점

이동 방식[1]:

1. 먼저 직선 방향(상하좌우)으로 한 칸 이동
2. 그 다음 대각선으로 한 칸 이동
3. 순 이동 거리: 정방향 1칸, 사선 1칸 (일반적인 체스의 나이트와 유사)

제약 사항:

- 경로상 첫 번째 칸(직선 방향)에 다른 기물이 있으면 이동 전체 불가
- 중간 기물이 있으면 해당 방향으로는 진출 불가
- 궁성 내 대각선은 일반 대각선 규칙 적용

이동 가능 위치: 2-8개 (현재 위치와 주변 상황에 따라 가변)

3.5 차 (Chariot/Rook) - 13점

이동 범위:

- 상하좌우 모든 직선 방향으로 원하는 만큼 이동 가능
- 궁성 내에서는 대각선 이동 가능 (다른 기물이 없을 경우)
- 가장 자유로운 이동성을 가진 기물

제약 사항:

- 다른 기물을 뛰어넘을 수 없음
- 경로상 다른 기물 직전까지만 이동 가능
- 궁성 내 대각선: 상, 마를 제외한 기물이 궁성 중앙에 있을 때만 이동 불가

이동 가능 위치: 최대 27개 (보드 크기와 현재 위치에 따라)

3.6 포 (Cannon) - 7점

이동 방식:

- **이동**: 차와 동일 (상하좌우 직선, 궁성 내 대각선)
- **캡처**: 반드시 기물 하나를 건너뛰어야만 상대 기물을 캡처 가능

캡처 규칙 상세:

1. 포가 어떤 기물 하나를 건너뜀 (아군 또는 적군 무관)
2. 건너뛴 기물의 다음 위치에 상대 기물이 있으면 캡처 가능
3. 건너뛴 기물 위치에는 갈 수 없음 (반드시 건너뜀)
4. 두 개 이상의 기물을 건너뛸 수 없음
5. 포 자신은 뛰어넘을 수 없음 (포와 포는 충돌)

제약 사항:

- 궁성 내 포는 건너뛸 기물이 있어야만 그 방향으로 이동 가능
- 기물이 없는 상태에서 포가 있는 경로는 통과 불가

3.7 졸 (Soldier/Pawn) - 2점

이동 방식:

1. 전진 방향으로 한 칸씩 이동 (초기 위치: y=6, 전진 방향: y 감소)
2. 좌우로 한 칸씩 이동 가능

제약 사항:

- 후진(뒤로 물러남) 불가
- 다른 기물을 건너뛸 수 없음
- 기물을 캡처할 때도 직진 또는 좌우 한 칸만 가능

특수 규칙:

- 졸이 승격(프로모션) 없음 (기본 점수 유지)

---

4. 게임 진행 규칙

4.1 턴 구조

한 턴의 진행:

1. 현재 플레이어가 자신의 기물 하나를 선택
2. 선택한 기물을 합법적인 위치로 이동
3. 상대 기물이 있는 위치로 이동하면 자동 캡처
4. 턴 종료 시 상태 확인 (체크, 체크메이트, 무승부)

5. 상대 플레이어로 턴 이동

선수 및 후수:

- 초(청/Blue): 선수 (게임 시작 시 먼저 움직임)
- 한(적/Red): 후수 (게임 시작 시 나중에 움직임)

4.2 체크 (Check/Janggun)

체크 정의: 상대의 궁이 다음 턴에 캡처될 수 있는 상태

체크 상황 처리:

- 플레이어가 체크 상황을 선언할 수 있음 (선택사항)
- 체크 상황에서 반드시 벗어나야 함
- 벗어나는 방법:
    1. 궁을 안전한 위치로 이동
    2. 공격 기물을 캡처
    3. 공격 경로에 방어 기물 배치

4.3 체크메이트 (Checkmate)

체크메이트 정의:

- 궁이 체크 상태에 있으면서
- 어떤 방법으로도 체크를 벗어날 수 없는 상태

게임 종료: 체크메이트 발생 시 상대 플레이어 승리

4.4 비장 (Bikjang) - 무승부

비장 상황:

- 두 궁이 같은 직선에서 서로 마주보고 있는 상태
- 중간에 다른 기물이 없어야 함

규칙:

- 비장 상황 발생 시 무승부 (비장)
- 현재 플레이어가 이 상황을 벗어나야 함 (벗어나지 않으면 패배)

4.5 스토우 (Stalemate/Pass)

스토우 규칙:

- 궁이 체크 상태는 아니지만 움직일 수 있는 합법적 수가 없는 경우
- 현재 플레이어는 턴을 패스할 수 있음
- 패스 후에도 상황이 변하지 않으면 무승부 (자장)

4.6 반복수 (Repeated Position)

반복 규칙[1]:

- 같은 기물 배치가 3회 반복되면 무승부 (만년장)
- 4회 반복되면 실격패로 처리

구현 방식:

- 보드 상태 해시 또는 기물 위치 기록으로 추적

---

5. 게임 종료 조건

5.1 승리 조건

1. **상대 궁 캡처**: 상대의 궁을 직접 캡처 (체크메이트)
2. **점수 승리**: 게임 중단 시 점수가 높은 쪽 승리

5.2 패배 조건

1. **궁 캡처**: 자신의 궁이 캡처됨
2. **기물 점수 차이**: 일정 턴 이후 점수 차이가 크면 자동 항복 권장

5.3 무승부 조건

1. **비장 (Bikjang)**: 두 궁이 마주보는 상태
2. **만년장 (Repeated Position)**: 3회 반복 (대한장기연맹 규정)
3. **자장 (Stalemate)**: 움직일 수 없는 상태에서 무승부 합의
4. **시간 초과**: 양측 모두 시간 초과 시

---

6. 점수 계산 (Scoring System)

6.1 기물별 점수

기물	개별 점수	설명
궁 (King)	∞	무한 점수 (캡처 시 패배)
차 (Chariot)	13	가장 강력한 기물
포 (Cannon)	7	특수 캡처 규칙
마 (Horse)	5	기동성 기물
상 (Elephant)	3	중간 수준
사 (Advisor)	0	방어 역할
졸 (Soldier)	2	기본 기물

Table 3: 기물별 점수표

6.2 게임 시작 점수

- **초 (Cho/Blue)**: 72점 (2 × 13 + 2 × 7 + 2 × 5 + 2 × 3 + 2 × 0 + 5 × 2)
- **한 (Han/Red)**: 73.5점 (후수 덤 1.5점)

6.3 점수 승리

조건:

- 게임이 정해진 턴 수에 도달하거나 기물 거의 모두 캡처된 상태
- 남은 기물의 총 점수로 승자 결정

---

7. 기술 사양

7.1 데이터 구조

보드 상태 표현:

class Board:
# 보드 크기: 9x10 (교차점)
SIZE_X = 9
SIZE_Y = 10

```
# 기물 저장: 2D 배열 또는 딕셔너리
# board[y][x] = Piece 객체 또는 None
pieces: dict[tuple[int, int], Piece]

# 게임 상태
current_turn: Player  # 현재 턴 플레이어
move_history: list[Move]  # 이전 수 기록
game_state: GameState  # PLAYING, CHECK, CHECKMATE, DRAW
```

기물 클래스 계층:

class Piece:
type: PieceType # KING, ADVISOR, ELEPHANT, HORSE, CHARIOT, CANNON, SOLDIER
player: Player # CHO (청) 또는 HAN (적)
position: tuple[int, int] # (x, y)
point: int # 기물 점수

```
def get_legal_moves(board: Board) -> list[tuple[int, int]]:
    # 기물별 이동 가능 위치 반환
    pass

def can_move_to(board: Board, target: tuple[int, int]) -> bool:
    # 특정 위치로의 이동 가능 여부 확인
    pass
```

7.2 규칙 엔진 (Rules Engine)

핵심 로직:

class GameEngine:
def **init**(self):
self.board = Board()
self.game_state = GameState.SETUP

```
def make_move(self, player: Player, from_pos: tuple, to_pos: tuple) -> bool:
    # 1. 기물 이동 가능 여부 확인
```

```
    # 2. 이동 수행
    # 3. 캡처 처리
    # 4. 게임 상태 업데이트 (체크, 체크메이트 등)
    # 5. 턴 이동
    pass

def is_check(player: Player) -> bool:
    # 특정 플레이어의 궁이 체크 상태인지 확인
    pass

def is_checkmate(player: Player) -> bool:
    # 특정 플레이어의 궁이 체크메이트 상태인지 확인
    pass

def is_bikjang() -> bool:
    # 비장 상태 확인
    pass

def get_game_state() -> GameState:
    # 현재 게임 상태 반환
    pass
```

7.3 UI 컴포넌트

프론트엔드 (React):

- **Board Component**: 9×10 보드 렌더링, 교차점 표현
- **Piece Component**: 기물 렌더링 및 선택
- **MoveHighlight Component**: 가능한 이동 위치 표시
- **GameInfo Component**: 현재 턴, 점수, 게임 상태 표시
- **MoveHistory Component**: 이전 수 기록 표시 및 회고
- **Timer Component**: 남은 시간 표시 (시간 제한 있는 경우)

7.4 백엔드 (FastAPI)

주요 엔드포인트:

- `POST /game/start` - 새 게임 시작

- `POST /game/{game_id}/move` - 수 실행
- `GET /game/{game_id}/state` - 현재 게임 상태 조회
- `GET /game/{game_id}/legal_moves` - 현재 플레이어 가능한 수
- `POST /game/{game_id}/resign` - 항복
- `GET /game/{game_id}/history` - 게임 기록 조회
- `POST /game/{game_id}/undo` - 수 되돌리기 (옵션)

---

8. 확장 기능

8.1 AI 대국

구현 방식:

- 미니맥스 (Minimax) 알고리즘 + 알파-베타 가지치기
- 평가 함수: 기물 점수 + 위치 가중치 + 전략적 포지셔닝
- 난이도 설정: 깊이 제한 (depth 3-6)

8.2 게임 분석 및 재생

- 기보 표기법 (예: "車 a0 → a5" 형식)
- 주요 수의 평가 제공
- 착각(미스) 부분 분석
- 게임 재생 기능 (턴-바이-턴)

8.3 온라인 멀티플레이

- WebSocket 기반 실시간 대국
- 게임 초대 및 매칭 시스템
- 플레이어 레이팅 시스템 (Elo 또는 Glicko)
- 채팅 및 사후평

8.4 챌린지 및 퍼즐 모드

- 특정 상황에서 최적의 수를 찾는 퍼즐
- 문제별 난이도 분류
- 해결 통계 및 보상 시스템

---

9. 테스트 계획

9.1 단위 테스트 (Unit Tests)

- 각 기물별 이동 규칙 검증
- 기물 캡처 로직
- 점수 계산
- 체크/체크메이트 판정

9.2 통합 테스트 (Integration Tests)

- 전체 게임 흐름 검증
- 복잡한 상황 시뮬레이션 (비장, 만년장 등)
- API 엔드포인트 테스트

9.3 수용 테스트 (Acceptance Tests)

- 사용자 스토리 기반 테스트
- UI/UX 테스트
- 성능 테스트 (초대형 게임 데이터베이스)

---

10. 개발 일정 (예상)

단계	예상 기간	주요 산출물
1. 규칙 엔진 개발	2-3주	기물 이동, 캡처, 게임 상태
2. 프론트엔드 UI 구현	2-3주	보드 렌더링, 기물 선택
3. 백엔드 API 개발	1-2주	게임 서버, 데이터 저장
4. 통합 및 테스트	2주	버그 수정, 최적화
5. AI 대국 기능	2주	기본 AI, 난이도 조절
6. 온라인 기능	2주	WebSocket, 매칭, 레이팅
7. 배포 및 모니터링	1주	프로덕션 배포
합계	12-15주	완성도 높은 게임 플랫폼

Table 4: 개발 일정 (대략적 추정)

References

[1] Korean Janggi Association (대한장기연맹). Retrieved from kja.or.kr and kingjanggi.or.kr, accessed February 2026. Rules including piece point values, palace setup, repeated position handling (3 repetitions = draw, 4 = loss).