Cahier des charges – Monopoly Vision Board

1. Présentation du projet

Nom du projet : Monopoly Vision Board
Objectif : Créer une version intelligente du jeu de plateau Monopoly utilisant une **caméra connectée (via Flask & OpenCV)** pour analyser le plateau en temps réel, détecter les pions, suivre l'évolution du jeu et afficher une **interface virtuelle synchronisée**.

L'application permettra :

- de suivre automatiquement les déplacements des pions réels,
- d'analyser la position des éléments sur le plateau,
- et d'afficher un **état numérique en direct** du jeu.

---

2. Fonctionnalités principales

Acquisition du flux vidéo

- [x] Connecter la caméra via OpenCV (`cv2.VideoCapture`)
- [x] Gérer la perte de connexion caméra (reconnexion automatique)
- [x] Flux MJPEG envoyé en continu vers Flask

Détection du plateau

- [ ] Détecter les **contours du plateau** avec `cv2.Canny` et `cv2.findContours`
- [ ] Corriger la **perspective** avec `cv2.getPerspectiveTransform`
- [ ] Identifier la **zone de jeu utile** et ignorer l'arrière-plan
- [ ] Reconnaître les **cases** du plateau via une grille (segmentation + homographie)
- [ ] Étalonner le système (référence visuelle du plateau Monopoly)

Reconnaissance des éléments

- [ ] Détecter les **pions** en couleur (HSV color filtering)
- [ ] Identifier la **position exacte** de chaque pion sur une case
- [ ] Détecter les **cartes Chance / Caisse de communauté** via `cv2.matchTemplate`
- [ ] Reconnaître les **maisons / hôtels** par analyse de forme (morphologie + contours)
- [ ] Détecter un **lancer de dés** (analyse de changement rapide + détection de points noirs)

Suivi et logique du jeu

- [ ] Associer chaque pion à un joueur
- [ ] Mettre à jour les positions sur le plateau virtuel
- [ ] Détecter les événements (passage sur une case spéciale, achat, prison, etc.)
- [ ] Sauvegarder l'historique du jeu (JSON ou base de données)
- [ ] Synchroniser les changements avec l'interface web

Affichage virtuel et interface web

- [ ] Créer une **vue Flask** affichant un plateau virtuel stylisé
- [ ] Superposer les **pions virtuels** à leur position réelle
- [ ] Afficher les **logs d'événements** (ex : "Le joueur 2 achète Rue de la Paix")
- [ ] Ajouter une section "Suivi de la partie" avec les cartes et soldes des joueurs
- [ ] Gérer les notifications visuelles (animation, couleurs dynamiques)

---

3. Traitements d'image OpenCV utilisés

Objectif	Méthode	Description
Détection des contours du plateau	`cv2.Canny`, `cv2.findContours`	Identifier la forme globale du plateau
Correction de perspective	`cv2.getPerspectiveTransform`, `cv2.warpPerspective`	Aplatir le plateau pour faciliter l'analyse
Segmentation des cases	`cv2.HoughLines`, `cv2.boundingRect`	Diviser le plateau en zones de cases
Filtrage couleur pour pions	`cv2.inRange`, `cv2.cvtColor(HLS/HSV)`	Isoler les pions selon leur couleur
Détection de formes	`cv2.approxPolyDP`, `cv2.contourArea`	Identifier les maisons, hôtels, dés
Suivi de mouvement	Frame differencing (`cv2.absdiff`)	Déterminer les déplacements entre frames
Template Matching	`cv2.matchTemplate`	Reconnaître les cartes Chance / Caisse
Stabilisation	`cv2.GaussianBlur`, `cv2.medianBlur`	Réduire le bruit pour de meilleures

Objectif	Méthode	Description
		détections

---

4. Interface utilisateur

Pages principales

- `/` → Flux vidéo et plateau virtuel
- `/dashboard` → Statistiques de la partie
- `/config` → Calibration du plateau

Éléments visuels

- Plateau virtuel animé (SVG ou Canvas)
- Pions affichés en surimpression
- Logs et notifications dynamiques
- Thème inspiré du **Monopoly original** (polices, couleurs, ambiance)

---

5. Outils et technologies

Domaine	Outil
Framework backend	Flask (Python)
Vision par ordinateur	OpenCV
Interface web	HTML / CSS / JS
Streaming vidéo	MJPEG
Données	JSON / SQLite (selon besoin)
Déploiement	Local ou serveur interne (Flask Dev Server)

---

6. Perspectives d'évolution

- Intégration d'un **mode multijoueur en ligne** (synchro Flask + WebSockets)
- Ajout de la **reconnaissance de texte** sur les cartes avec Tesseract (OCR)

- Utilisation d’un modèle d’IA pour **identifier automatiquement les éléments** du plateau