UNIVERSITÉ DE LORRAINE | IUT nancy Charlemagne Informatique

Etude préalable robot billard

Antoine FONTANEZ - Nathan PIERROT - Corentin FROGER

Tuteur : Pierre-André GUENEGO

Vision du produit

Notre projet consiste en le développement d'un robot capable de jouer au billard de manière fluide et réfléchie. Il sera accompagné d'un simulateur permettant de créer des scénarios de tests. Ce travail va permettre de présenter un exemple concret de projet réalisé en BUT informatique et les compétences acquises en fin de formation lors des portes ouvertes de l'IUT, le samedi 1er mars 2025.

Le projet est novateur, bien que plusieurs robots jouant au billard ont déjà été créés par le passé, aucun ne ressemble vraiment au robot que nous développons. En effet, nous utilisons un robot roulant alors que la plupart des robots joueurs de billard existants possèdent un bras articulé et utilisent une queue.

Liste des fonctionnalités du système

- Faire communiquer la page web avec le robot grâce à une connexion wifi
- Faire bouger le robot depuis l'interface web
- Associer la caméra au site web
- Visionner les images de la table de jeu prises par la caméra sur le navigateur web
- Analyser les objets que la caméra voit :
    - Les boules
    - Le tag ArUco (donc par extension le robot)
    - Les trous dans lesquels pousser les boules
- Permettre la simulation de scénarios :
    - Jouer au billard
    - (Autres à voir plus tard)
- Afficher les trajectoires prévues pour le robot
- Actualiser les trajectoires en fonction du déplacement réel du robot
- Lancer une simulation du comportement du robot sur la table de jeu :
    - Dans des configurations prédéfinies
    - Permettre à l'utilisateur de bouger lui même les éléments de la simulation (boules et robot)
- Lancer une simulation en fonction des positions réelles des boules et du robot (grâce à la caméra)
- Changer la vitesse de la simulation
- Importer une configuration à partir d'un fichier
- Sauvegarder une configuration dans un fichier

Recensement et évaluation des risques

Tout d'abord nous allons devoir apprendre à faire de l'arduino pour pouvoir faire bouger le robot. Cela peut s'avérer être une tâche plutôt complexe puisque nous allons devoir faire du C (utilisé par l'arduino), un langage dont nous n'avons pas l'habitude à cause de son bas niveau. Heureusement que c'est uniquement une petite partie du projet, mais elle arrive dès le début. **Niveau de difficulté estimé : 7/10**

De plus, nous allons surement rencontrer des problèmes de connexion réseau. Comment gérer les potentielles déconnexions du robot ? Il va falloir que nous mettions en place des sécurités (permettre au serveur de continuer à tourner normalement même si le robot n'est plus là) et faire en sorte que le robot se reconnecte automatiquement au réseau. Tout en prenant en compte qu'il devra toujours exécuter les ordres qu'on lui donne. **Niveau de difficulté estimé : 8/10**

En parlant d'ordres donnés au robot, nous allons devoir faire en sorte qu'il exécute le dernier ordre reçu, afin de rester cohérent dans ses mouvements. Il faut aussi mettre en place un système qui lui permet de connaître l'heure d'envoi d'un ordre, afin de savoir s' il est toujours valable ou pas (s'il doit l'exécuter ou non). **Niveau de difficulté estimé : 8/10**

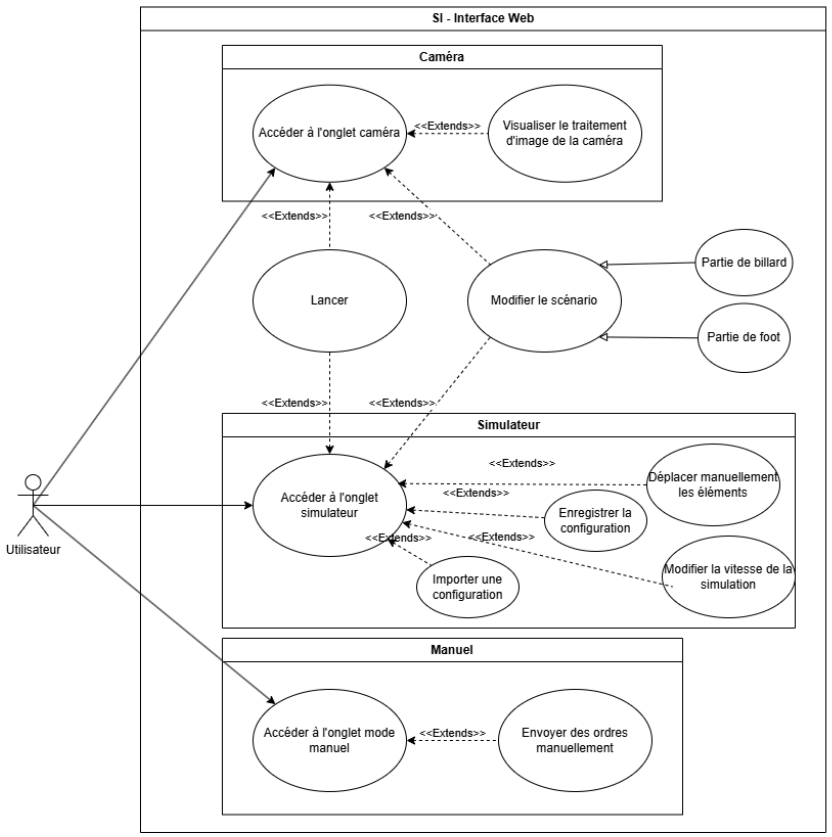
Ensuite nous allons sûrement faire face à des petits conflits dans l'équipe sur des questions de conception ou de réalisation. Nous utilisons tous un vocabulaire différent ce qui mène à des confusions, alors que parfois nous parlons de la même chose. Il faut que nous essayons de nous exprimer de la manière la plus claire possible mais également de comprendre et de reformuler les explications de nos coéquipiers pour vérifier que nous avons correctement compris ce qu'il voulait nous expliquer. **Niveau de difficulté estimé : 5/10**

Un autre souci serait de bloquer sur une étape de la réalisation du projet (par exemple faire en sorte que le robot se déplace de manière fluide). Nous allons donc devoir travailler tous ensemble sur le code afin d'essayer de trouver une solution acceptable et permettre à tout le monde de retourner travailler sur sa fonctionnalité. **Niveau de difficulté estimé : 6/10**

Justement : la fluidité du robot. Nous savons que ce point à posé problème l'année précédente, nous allons donc y porter une attention particulière. Cela nous prendra surement du temps avant de trouver une solution convenable. Nous allons donc probablement travailler tous ensemble sur la fonctionnalité afin de trouver une solution plus rapidement et de ne pas s'éterniser sur la question. **Niveau de difficulté estimé : 8/10**

Une grosse partie du projet est la réalisation d'un simulateur, cela va nous prendre beaucoup de temps. Il faut que nous réussissions à implémenter une physique réaliste, qui représente le plus fidèlement possible la réalité. Cela est indispensable si nous voulons substituer la caméra par une simulation. **Niveau de difficulté estimé : 7/10**

Diagramme de cas d’utilisation



Préconditions

Le robot doit être allumé, connecté au réseau et on doit pouvoir lui envoyer des ordres via le site Internet

Postconditions

Les conditions du scénario choisit par l'utilisateur sont remplies

Étapes du déroulement normal

L'utilisateur :

- Visualise le traitement d'image de la caméra
- Accède au simulateur
- Modifie le scénario par défaut
- Lance la simulation

Variantes

L'utilisateur à envoyé quelques ordres au robot avant de lancer le scénario

L'utilisateur a laissé le scénario par défaut

Contraintes non fonctionnelles

Un ordre qui arrive avec plus d'une seconde de retard sera ignoré
Si le robot perd la connexion Wi-Fi, il doit se reconnecter automatiquement

Diagramme d'État

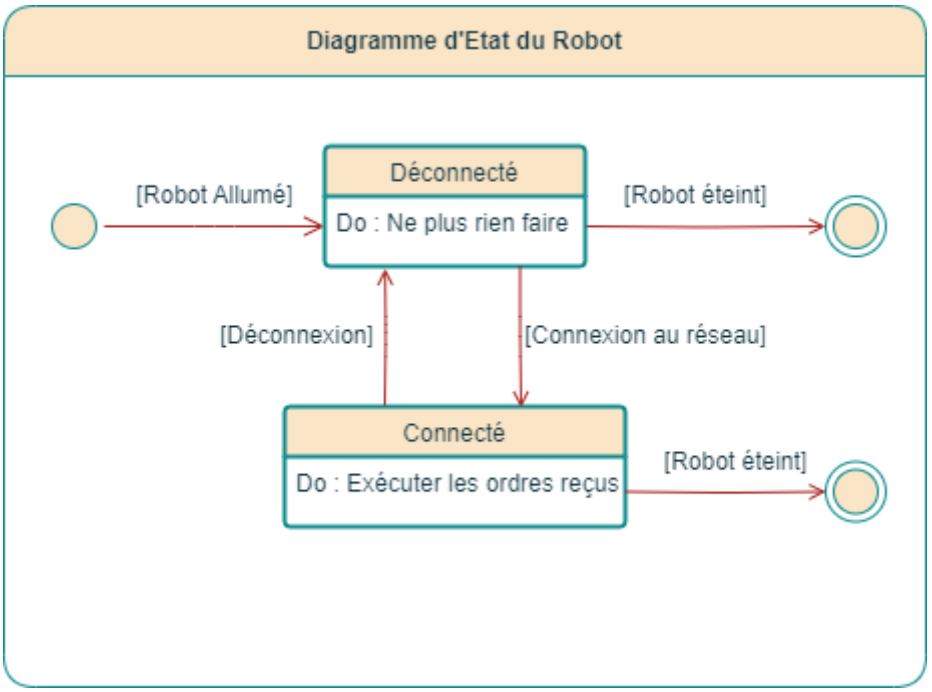


Diagramme de classes

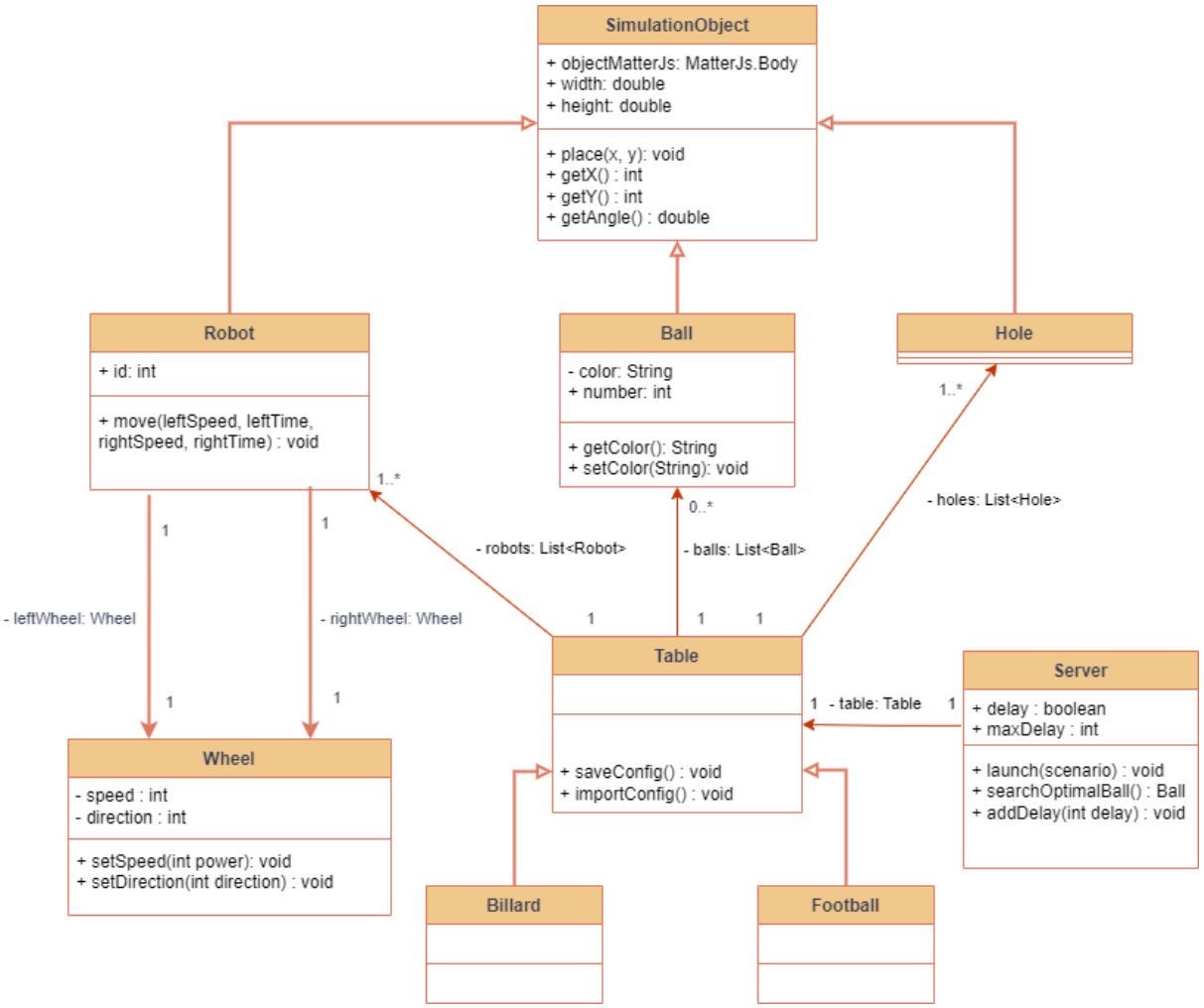


Diagramme de séquence système

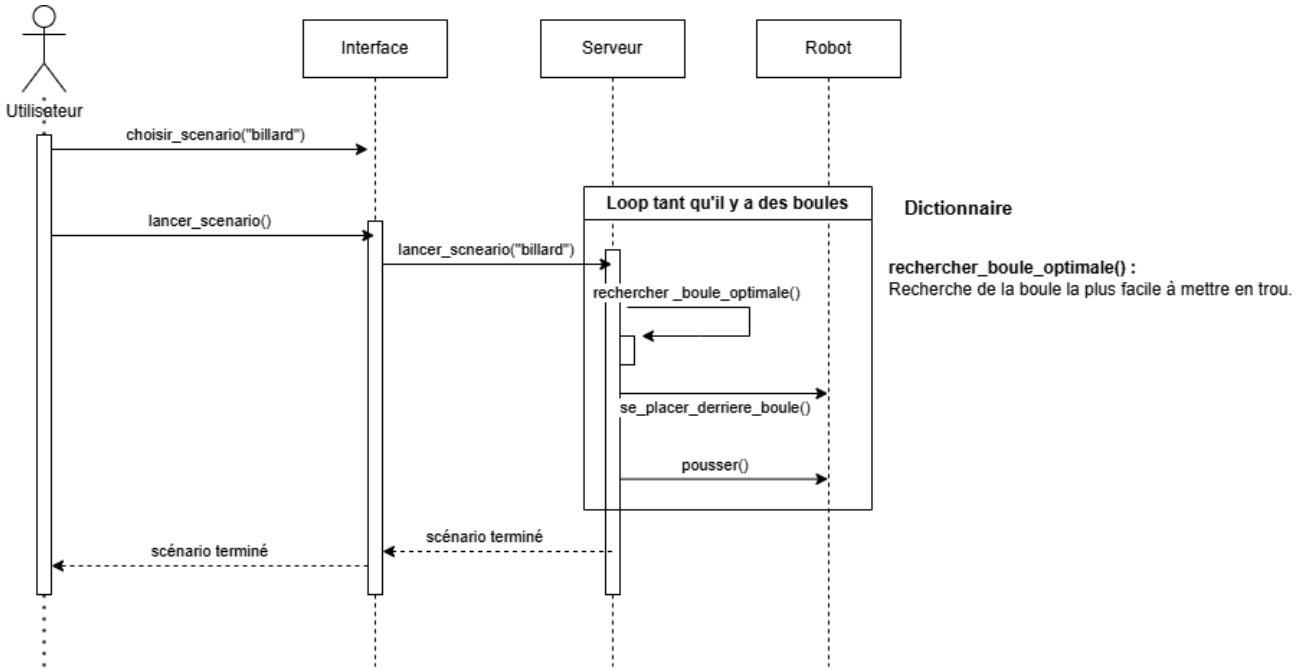
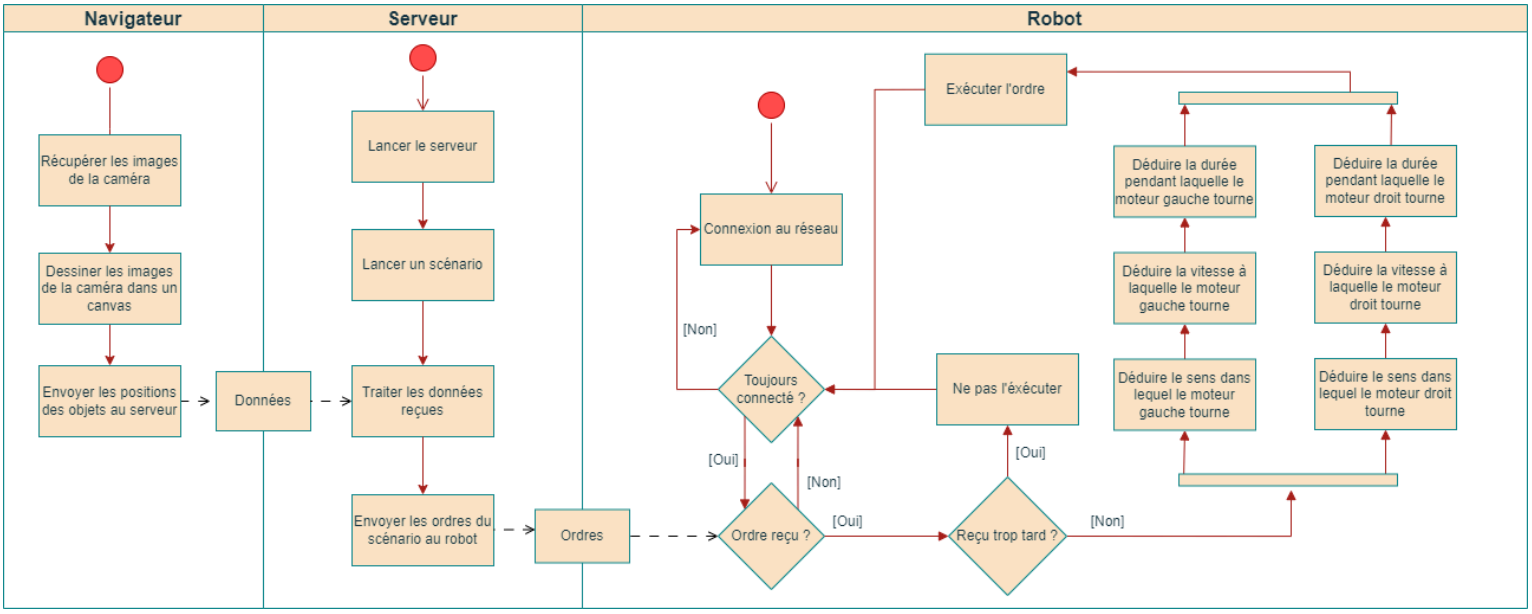
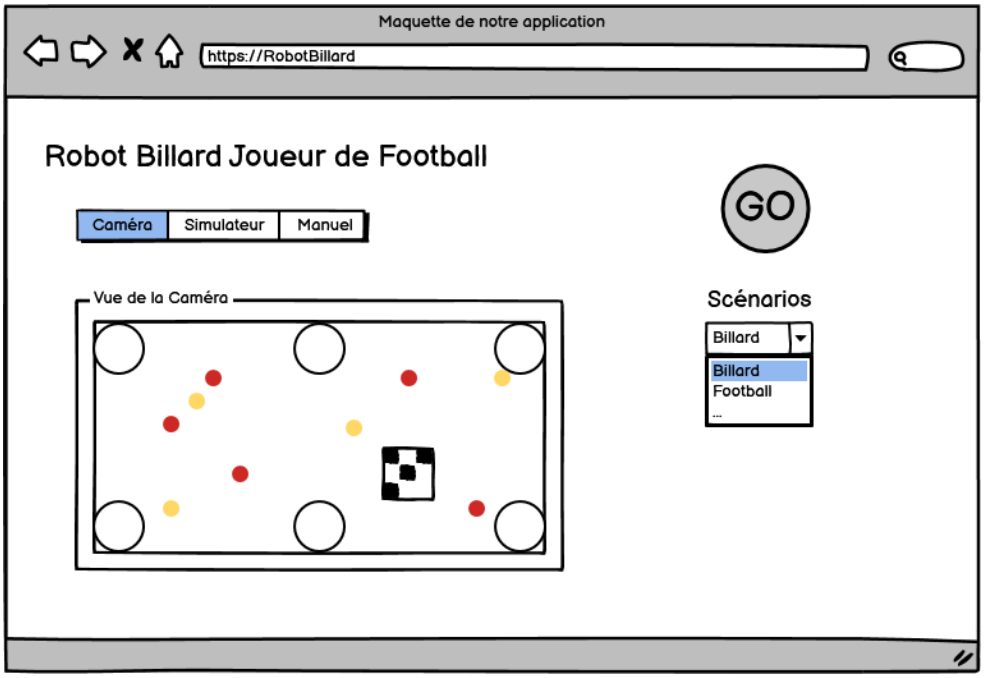


Diagramme d’activité

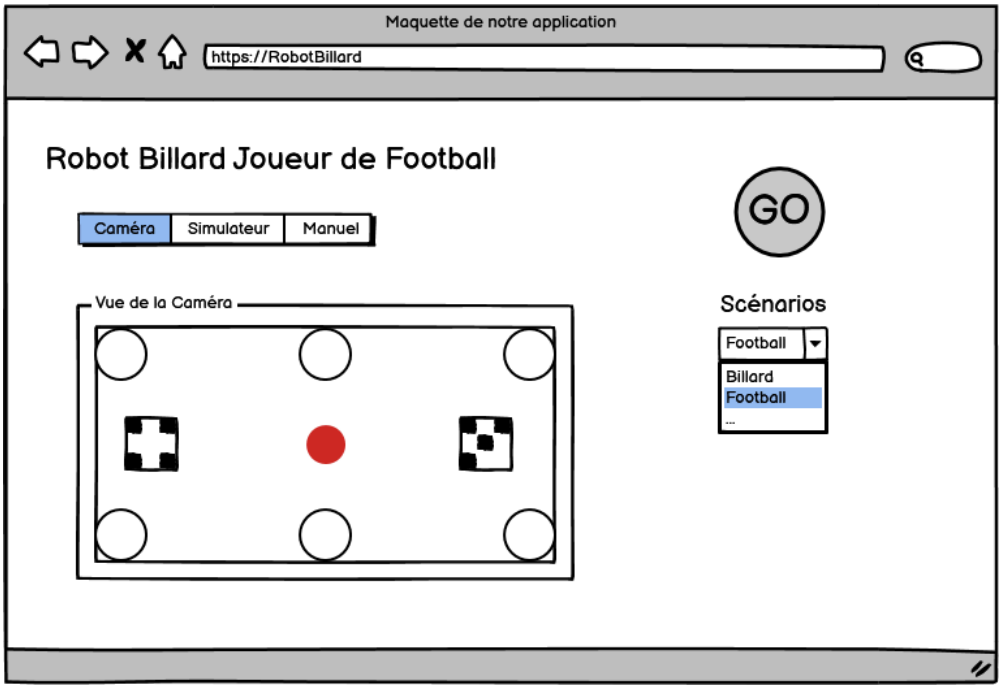


Maquettes

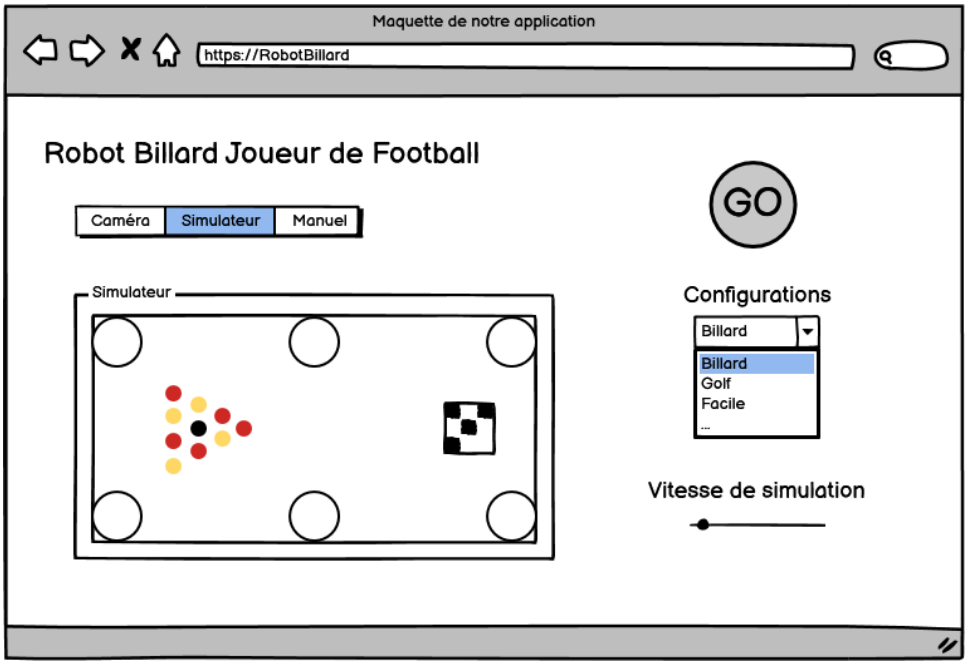
Vue de la caméra, scénario : billard



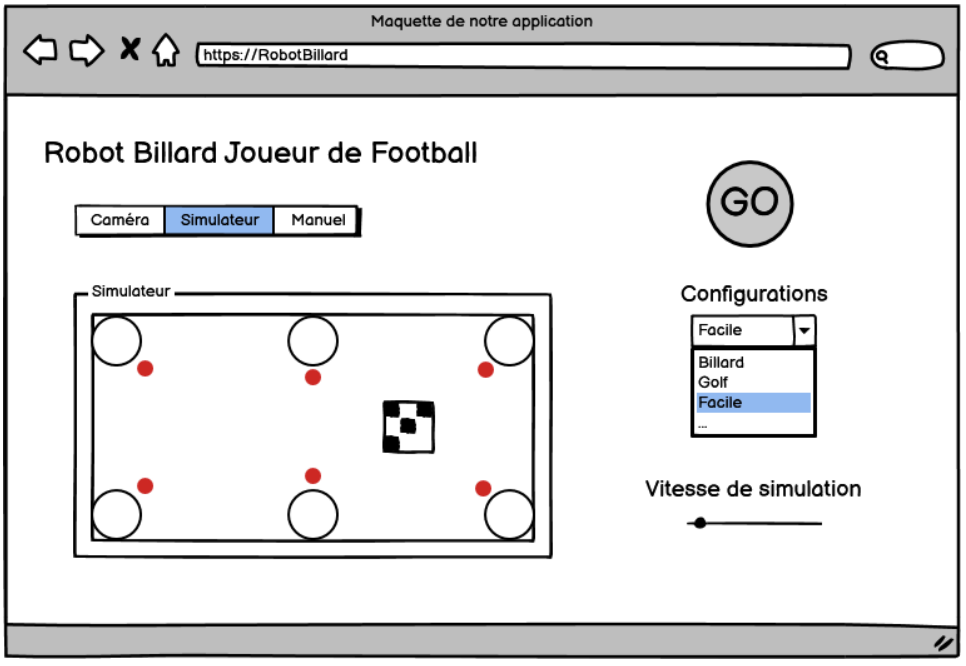
Vue de la caméra, scénario : football



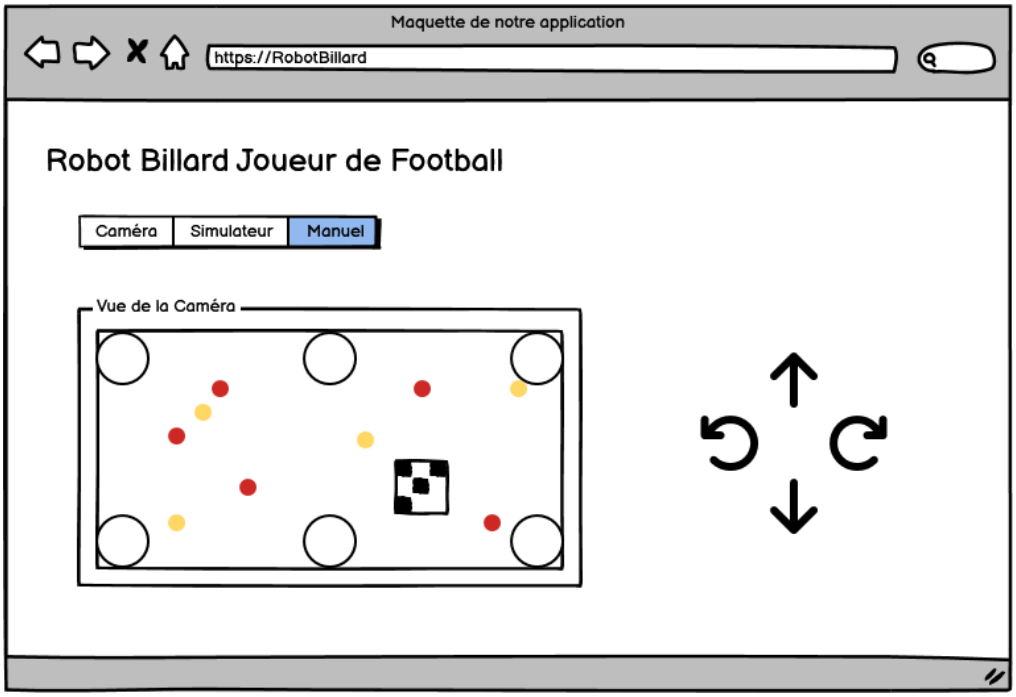
Vue du simulateur, configuration : billard



Vue du simulateur, configuration : facile



Vue de la caméra, mode commandes manuelles



Planning

Itération 1 :

- Prototype de simulateur (Antoine) :
    - Affichage des objets (robot, boules, troues) dans le canvas
    - Possibilité de déplacer les objets à la souris
- Prototype de l'interface web (Nathan)
- Afficher les images de la caméra dans un canvas HTML (Corentin / Nathan)
- Reconnaître les boules grâce à OpenCV (Corentin / Nathan)
- Contrôler la vitesse du robot manuellement (modifier la puissance des moteurs) depuis un navigateur (Corentin / Antoine)
- Connecter le robot au réseau (Antoine / Corentin / Nathan)

Itération 2 :

- Reconnaître l'ArUco du robot (Corentin / Nathan)
- Terminer l'interface web (Nathan)
- Refaire le serveur node.js (Corentin)
- Faire bouger le robot dans le simulateur (Antoine)
- Configurations pré faites dans le simulateur (Antoine)

Itération 3 :

- Gestion de la latence (Antoine / Corentin / Nathan)
- Programmer au moins un scénario en entier dans le simulateur (Antoine)
- Différencier les boules de billards des trous (Corentin / Nathan)

Itération 4 :

- Programmer les calculs de trajectoires du robot pour qu'il puisse bouger de manière fluide (Corentin / Nathan)
- Rendre paramétrable la vitesse de déroulement de la simulation (Antoine)

Itération 5 :

- Corriger les possibles bugs (Antoine / Corentin / Nathan)
- Programmer un scénario réel en entier (Antoine / Corentin / Nathan)

Itération 6 :

- Corriger les possibles bugs (Antoine / Corentin / Nathan)

Itération 7 :

- Corriger les possibles bugs (Antoine / Corentin / Nathan)