# Petites explications sur les systèmes de coordonnées :

Dans le script du billard, il y a trois systèmes de coordonnées distincts qui doivent interagir. Il est donc primordial de les connaitre et de connaitre leur relations.

Tout d’abord il y a les coordonnées en pixels de la bille sur l’image vu par la caméra. Ce repère a pour origine l’angle supérieur gauche de l’image. Selon la caméra utilisée et son placement ces coordonnées peuvent être très différentes pour une même position de la balle (par exemple deux caméras placées au même endroit mais avec des résolutions différentes attribueront une position différente à la balle). Ces coordonnées sont celles déterminées par l’algorithme de tracking.

Vient en second temps les coordonnées « réelles » de la balle, celles dans l’espace (le plan ici, le billard pouvant se ramener à un problème à deux dimensions, avec une vue de dessus de la table). Le repère de ce système de coordonnées est situé au centre de la table de billard et a pour unité le mm. La connexion entre le premier repère et le deuxième nécessite le calcul d’une homographie entre les deux plans (de l’image de la caméra et de la vue de dessus de la table).

Enfin le dernier système de coordonnées est celui de l’image projetée par le vidéo projecteur. Il a pour origine le coin supérieur gauche de l’image et ses dimensions dépendent de la taille de l’écran (qui est affiché sur la table par le vidéoprojecteur). Il doit être possible de faire correspondre la position réelle de la balle à un point sur l’écran de manière à ce qu’il soit projeté sur la position réelle de la balle par le vidéo projecteur.

# Présentation du script

Lorsque le script se lance, l’algorithme projette une image blanche avec un rectangle noir et demande à l’utilisateur de placer l’intérieur de la table de billard (partie verte entre les bords) dans le rectangle. Les dimensions de ce rectangle ont été calibrés pour correspondre à la taille de la table pour la position du vidéoprojecteur. Ce qui m’amène à préciser une première hypothèse fondamentale : **le vidéoprojecteur a une position fixe et bien définie au-dessus de la table de billard**. Le placement de la table est primordial pour assurer que les coordonnées réelles de la bille et du point la représentant sur l’image affichée par le vidéoprojecteur est coïncident.

Une fois que la table est placée, l’utilisateur doit appuyer sur « q » pour fermer l’image affichée et passer à la suite. Attention, pour que l’on puisse lire les touches pressées par l’utilisateur, il faut qu’une fenêtre OpenCV du PC soit sélectionnée. Au besoin, cliquer sur l’image avant d’appuyer sur « q ». A partir de cet instant, on fait l’hypothèse que **la table ne bouge plus.**

Une fois la table placée, la correspondance entre le deuxième système de coordonnées et le troisième est effectué. On va maintenant s’intéresser à la correspondance entre les deux premiers.

Le but de cette partie est de trouver une homographie entre le plan de la caméra et celui de la vue de dessus de la table. A cet instant un choix s’impose à l’utilisateur : utiliser la matrice d’homographie sauvegardée à l’extérieur du programme dans le fichier « AllData.json » (par exemple si il a déjà trouvé l’homographie la dernière fois qu’il a lancé le script, il n’a pas besoin de la déterminer à nouveau) ou la déterminer à l’aide d’une fonction que nous avons créé. Elle nécessite l’usage de la caméra qui, étant externe au PC, met un certain temps à se connecter, il faut donc être patient ! Lorsqu’une image venant de la caméra s’affiche, c’est que vous pouvez commencer. L’algorithme demande à l’utilisateur de placer la boule rouge dans le coin inférieur gauche de la table puis de valider avec la barre d’espace ou la touche entrée. Le coin inférieur gauche est repérable par la position de la plaque dorée en bas de la table de billard (voir Figure 1). **Attention !** L’algorithme de détection fonctionne par recherche de couleur, dans la mesure du possible, enlever tout autre objet rouge du champ de vision de la caméra. Une fois le premier angle détecté, l’algorithme demande à l’utilisateur de placer la boule dans les trois derniers angles.  
Connaissant ainsi la position de chacun des angles da la table dans le plan de la caméra et dans le système de coordonnées « réelles », on peut déterminer une homographie entre les deux plans. Si le programme ne parviens pas à trouver d’homographie, c’est que la position des angles a été mal détectée, on doit alors recommencer la procédure (tracking de la position des 4 coins).  
Pour vérifier que tout a bien fonctionné, on demande à l’utilisateur de retirer la boule et on prend une nouvelle image de la table (après validation avec espace ou entrée) à laquelle on applique l’homographie déterminée. Si tout c’est bien passé on obtient une vue de dessus de la table. L’utilisateur peut alors confirmer si l’homographie trouvée est correcte avec « y » ou l’infirmer avec « n » auquel cas on recommence la procédure (tracking de la position des 4 coins).



Figure : Position des coins de la table

Arrivé à cette étape, toutes les procédures préliminaires ont été effectuées, on peut rentrer dans le cœur du programme.

Le programme comporte deux modes principaux, le jeu libre et la capacité de rejouer des coups (séparé en 3 étapes). Au tout départ, nous sommes dans le mode jeu libre. On affiche sur l’écran du PC (et donc sur la table de billard et à côté) la fenêtre du fond sur laquelle on peut tracer et écrire des informations, elle est donc en permanence modifiée mais s’appuie sur un modèle de base (Figure 2). C’est une sorte d’interface graphique avec l’utilisateur. Dans le premier mode (appelé mode 0 dans la suite), on affiche la trajectoire de la balle en temps réel (elle s’estompe en 3 secondes). Sur la zone d’information à droite, on affiche si la balle est en mouvement ou à l’arrêt. Sur la zone den dessous de la table, on indique à l’utilisateur les commandes disponibles, à savoir : sauvegarder la dernière trajectoire de la bille (elle doit être à l’arrêt) en appuyant sur « s », entrer dans le fonctionnement type rejeu, en commençant par le mode 1, avec « r » ou quitter l’application avec « q ».

Une image contenant texte

Description générée automatiquement

Figure : Image de base projetée sur le billard

Les autres modes, qui participent à la fonction de rejeu, n’ont pas encore été implémentés, mais voici ce qui est prévu pour leur fonctionnement :

Le mode 1 permet de choisir la trajectoire à rejouer. On affiche donc sur l’interface graphique les numéros de l’ensemble des coups enregistrés. L’utilisateur doit alors cliquer sur un numéro pour choisir le coup qu’il veut rejouer. *// On se borne à 9 coups où alors on implémente un double waitKey avec une validation par entree pour pouvoir enregistrer plus de coups ?* Une fois le choix fait, on entre directement dans le mode 2. Il est sinon possible de revenir au mode 0 avec « échap ».

Le mode 2 attends de l’utilisateur de replacer la bille à sa position initiale. Pour cela, on affiche directement sur le billard la position où la bille doit être placée par un cercle. En parallèle, on tracke la position actuelle de la balle. Lorsque elle a été bien placée pendant 2 secondes, on confirme à l’utilisateur que la balle a été correctement placée par un signal visuel ou sonore. On entre alors automatiquement dans le mode 3. Il est sinon possible de revenir au mode 0 avec « échap ».

Le mode 3 constitue la véritable phase de rejeu. La trajectoire enregistrée est affichée en pointillés, el on affiche la nouvelle trajectoire de la balle en traits continus, sans qu’elle ne s’estompe. L’utilisateur peut alors essayer de refaire le même coup pour le travailler et comparer l’écart entre les deux trajectoires ou tenter un tout autre coup si celui qu’il avait fait ne le satisfaisait pas.

// Pos 1

Lorsque la balle est arrêtée, on revient automatiquement au mode 0.

// Pos 2

Lorsque la balle est arrêtée, le programme est en stand-by et attends que l’utilisateur lui indique quoi faire. Les trajectoires sont pendant ce temps toujours affichées, ce qui laisse la possibilité à l’utilisateur de les étudier aussi longtemps qu’il le veut. Pour continuer à jouer l’utilisateur a de nombreuses possibilités : il peut retourner dans le mode qu’il souhaite en appuyant sur « 0 » (il a terminé de s’entrainer sur ce coup et souhaite poursuivre le jeu), « 1 » (il a terminé de s’entrainer sur ce coup et souhaite en travailler un autre) ou « 2 » (il souhaite continuer à rejouer ce même coup). Il lui est aussi possible de sauvegarder la nouvelle trajectoire, en écrasant l’ancienne avec « Maj + S » ou en la sauvegardant dans un nouveau slot avec « s ».