TME numéro 9

Exercice 1 Écrivez une fonction `mise_a_jour_pos_balles : unit -> unit` qui met à jour la position de toutes les balles se trouvant dans l'espace de jeu (en utilisant les vitesses et les directions stockées dans les enregistrements de la variable globale `balles`). Lorsque celles-ci sortent de l'espace de jeu ou bien lorsqu'elles rentrent en collision avec un cube, elles sont supprimées de la liste des balles. Vous pourrez supposer que, lorsque cette fonction est appelée, un mutex a été locké. N'oubliez pas d'enregistrer votre fonction auprès du serveur.

Exercice 2 Écrivez une fonction `mise_a_jour_balles : unit -> unit` qui met à jour la position des balles (en appelant la fonction précédente) et détermine celles qui doivent exploser sur des vaisseaux. Dans ce dernier cas, le nombre de points de vie du vaisseau diminue et, s'il n'en reste plus, il est détruit. Vous pourrez supposer que, lorsque cette fonction est appelée, un mutex a été locké.

Exercice 3 Écrivez une fonction `mise_a_jour_vaisseaux : unit -> unit` qui met à jour les positions des vaisseaux en cours de partie. Pour chaque vaisseau, calculez la nouvelle position de ce dernier en utilisant sa vitesse et sa direction. Si le vaisseau n'entre en intersection avec aucun autre vaisseau ni aucun obstacle, vous pouvez passer au vaisseau suivant. Sinon, il faut trouver le meilleur déplacement possible au sens de celui qui permet d'aller le plus loin possible dans la direction la plus proche possible de celle suivie par le vaisseau tout en évitant les intersections avec les obstacles et les autres vaisseaux spatiaux (cf. l'algorithme que nous avons vu en cours). Vous pourrez supposer que, lorsque cette fonction est appelée, un mutex a été locké.

Exercice 4 Arrivé à ce stade, vous avez en main toutes les briques vous permettant de mettre à jour les positions des balles et des vaisseaux spatiaux de l'espace de jeu, ainsi que de gérer les explosions des balles (sur les obstacles et les vaisseaux spatiaux). Bien entendu, ces opérations de mise à jour doivent être effectuées régulièrement et ce indépendemment de ce que font les clients connectés au serveur. Il est donc logique de penser que le serveur lance une *thread* qui s'occupera uniquement de ces mises à jour. Cette thread tournera en concurrence avec les clients connectés.
Écrivez une fonction `mise_a_jour : unit -> unit` correspondant à la fonction exécutée par cette thread : tous les 40ème de seconde, la fonction met à jour les balles et les vaisseaux spatiaux de l'espace de jeu (positions + explosions). Comme ces opérations s'effectuent en concurrence avec les clients, il faut mutexer les opérations sensibles. Dans le fichier `aide_serveur`, le mutex utilisé par les clients pour accéder aux informations des listes `balles` et `vaisseaux` s'appelle `mutex : Mutex.t`.