Rapport du Système

Membres du groupe : Jiangnan HUANG , Xiaoxuan HEI

Le projet Système est de programmer un jeu vidéo “ snake ” en C avec SDL.

Une vue du jeu

Makefile

Pour compiler le programme assez facilement, on a fait un Makefile :

```
prog:main.c

	g++ -o prog main.c -lSDL2 -lSDL2_mixer
```

Partie A : Améliorations

Pour faire un jeu vidéo, évidement il faut actualiser l’écran assez fréquemment, donc on utilise une boucle infinie avec un petit delay (fonction SDL_Delay() pour fixer le dur temps) . Dans chaque boucle le programme va détecter est-il y a un bouton qui est appuyé sur le clavier.

Pour l’Introduction, on a utilisé la fonction SDL_SetWindowFullscreen() dans SDL. Si le programme fonctionne en mode fenêtre, et il a détecté que le bouton Q a été appuyé, il va appeler cette fonction avec les arguments (window, SDL_WINDOW_FULLSCREEN) pour passer en mode plein écran. Par contre si le programme est en mode plein écran et il a détecté le bouton W, il va l’appeler avec les arguments (window, 0) pour passer en mode fenêtre.

```
}  else if(state[SDL_SCANCODE_Q]){
        SDL_SetWindowFullscreen(window, SDL_WINDOW_FULLSCREEN);

    }
    else if(state[SDL_SCANCODE_W]){
        SDL_SetWindowFullscreen(window, 0);
    }
```

Pour étape 1, on ne veut pas mourir dès qu'on touche la frontière, pour ne pas mourir, on a modifié un peu les codes dans la partie update(), quand le programme directe que la position de notre tête du serpent a dépassé la frontière, on n'appelle pas la fonction gameover() mais transmettre la tête à la côté opposé (la direction de la tête ne change pas), avec cette façon on peut réaliser la modèle tonique.
Par exemple if (head.x > MAX_X) on fait { head.x = 0; }.

```
if (head.x > MAX_X) {
    head.x = 0;
}
if (head.x < 0)     {
    head.x = MAX_X;
}
if (head.y > MAX_Y) {
    head.y = 0;
}
if (head.y < 0)     {
    head.y = MAX_Y;
}
```

Pour l'étape 2, en fait on n'a pas réussi à faire un modèle continu. On l'a essayé et on a arrivé à déplacer le serpent dans toutes les directions (enlever TILE_SIZE , mettre la tête de serpent plus grande qu'avant pour entourer plusieurs pixels, la direction tourner un petit peu avec chaque manipulation.) mais malheureusement on a été bloqué par la détection du clavier. Comme le delay n'a pas changé et la tête est devenu plus grande, il semble que la vitesse du serpent a réduit, donc on a essayé de diminuer encore le delay, mais avec cette méthode la détection du clavier devient trop vite, et la manipulation du jeu n'était plus agréable (ça devient impossible plus précisément). Avec beaucoup de réflexion, on n'a pas finalement trouvé une solution. Par contre on a utilisé une autre façon de rendre le déplacement plus sympa : on a changé les 4 directions "up", "down", "left", "right" par 8, "E", "SE", "S", "SW", "W", "NW", "N" et "NE". Dans chaque détection la programme va détecter la direction actuelle de la tête, et en plus décider la direction prochaine avec le bouton appuyé. Le serpent tourne dans le sens des aiguilles d'une montre si on a appuyé LEFT, le sens contres si RIGHT. Par notre observation, le serpent peut tourner assez courament.

Pour l'étape 3, comme on n'a pas fait le mode continu, on n'a pas besoin de considérer les cellules du serpent comme des cercles. Mais on a créé quand même un Array position[] pour garder en mémoire toutes les positions de chacun des cellules du serpent. Avec les positions on

a créé une autre règle : le serpent va avaler lui-même quand sa tête touche la queue, toutes les cellules après la cellule qui est en collision vont disparaître.

Partie B : extensions

1. On met des mines sur la fenêtre, les positions de mines changent automatiquement dans un délai de 20 delay. Si le serpent heurte une mine, il va être tué. (En fait, ce n'est pas possible de différer les deux situations « si sa tête y passe » ou « son corps y passe », parce que le corps ne passe que la zone que la tête passe.)
2. On met 4 types de fruits avec différentes textures. Si le serpent mange un fruit, la longueur +1. On met aussi des fantômes sur la fenêtre. Si le serpent les touche, la longueur du serpent -1. Quand il n'y a que la tête, le jeu termine ici.
3. On met des murs qui occupent de 20 pixels. En jugeant la position du serpent et la comparant avec les positions des murs, on peut savoir si le serpent touche les murs ou pas. Si le serpent les touche quand il n'y a que la tête, il va rebondir sans condition. Si la longueur plus de 1, quand il touche les murs dans les directions de l'est, de l'ouest, du nord, et du sud, il va être tué. Mais des autres directions, il va rebondir.
4. Le serpent peut aussi lancer les missiles en contrôlant la vitesse du serpent. C'est-à-dire, le missile marche 1 pixel dans 1 delay, mais le serpent marche 1 pixels après 5 delay. Après que le serpent lance un missile, si le missile passe fruit, la longueur +3. Si le missile passe le fantôme, la longueur +1. Si le missile touche les murs, il va disparaître.

5. On a ajouté les différents sons en utilisant SDL_Mixer: quand il mange un fruit ; quand il touche une mine ; quand il heurte un mur ; quand il touche un fantôme ; quand il lance un missile ; quand le missile touche un mur et quand le jeu termine. Pour étendre le son de terminaison, on ajoute un sleep(3) avant exit(0). On a réussi d'ajouter un fond sonore, mais on pense que c'est un peu bruyant donc on l'a annulé.
6. On a ajouté les textures de la tête pour adapter les différentes directions de déplacement. Avec les têtes de nord-est, de sud-est, de nord-ouest, de sud-ouest etc, le serpent a l'air plus réaliste en bougeant.

Partie C : améliorer le gameplay

On a essayé de faire le deuxième serpent en utilisant deux threads mais on n'a pas arrivé. On va continuer d'apprendre la connaissance sur ce sujet.

Utilisation :

1. Appuyez sur « Q » pour mode plein écran et appuyez sur « W » pour revenir en mode fenêtré.
2. Appuyez sur le bouton gauche de la souris pour une pause et appuyez sur le bouton droit de la souris pour recommencer.
3. Appuyez sur « up » pour accélérer et appuyez sur « down » pour ralentir. Appuyez sur « left » et « right » pour changer sa direction.
4. Appuyez sur « A » pour lancer un missile.

Bug:

1. Quand on passe en mode plein écran et quand on revient en mode fenêtré, un texture de la tête ou corps va apparaître. Nous avons essayé de mettre un pop_tail() avant changer son mode, mais ça ne marche pas.
2. Comme on a fait, que quand le serpent entre en collision avec son corps, il va manger les corps après. Mais s'il y a 8 corps qui consiste un cercle, quand sa tête mange sa queue, quelques corps va apparaître sur l'écran et rester là-bas.
3. Quand le serpent heurte les murs, il va rebondir. Mais si vous changez sa direction tout de suite, il va manger le mur. Quand le serpent marche à coté des murs, il ne rebondit pas. Mais si vous changez sa direction tout de suite, il va manger aussi le mur.
4. Si on lance deux missiles, après le lancement du deuxième missile, le premier missile ne se déplace plus.