HAL
open science

Des jeux vidéo pour l'enseignement du langage C

Sophie Dupuis, Didier Crestani, Vincent Creuze, Bertrand Gelis, Eric Pommier, Vincent Thomas, Isabelle Pinon

To cite this version:

Sophie Dupuis, Didier Crestani, Vincent Creuze, Bertrand Gelis, Eric Pommier, et al.. Des jeux vidéo pour l'enseignement du langage C. CETSIS: Colloque sur l'Enseignement des Technologies et des Sciences de l'Information et des Systèmes, May 2017, Le Mans, France. 2017. lirmm-01522447

HAL Id: lirmm-01522447

https://hal-lirmm.ccsd.cnrs.fr/lirmm-01522447

Submitted on 15 May 2017

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Des jeux vidéo pour l'enseignement du langage C

Sophie Dupuis, Didier Crestani, Vincent Creuze,
Bertrand Gelis, Eric Pommier, Vincent Thomas et Isabelle Pinon
sophie.dupuis@umontpellier.fr
IUT de Montpellier, Département GEII, 99 avenue D'Occitanie, 34296 Montpellier, France

RESUME : Cet article présente deux séances de Travaux Pratiques (TPs) du module Informatique du premier semestre de DUT Génie Electrique et Informatique Industrielle (GEII). Afin de susciter l'intérêt des étudiants et de les motiver dans l'apprentissage du langage C, ces TPs ont pour but de développer une version du jeu vidéo Pong, proposé par la société Atari en 1972.

Mots clés : Informatique industrielle, programmation, langage C, jeu vidéo, librairie SDL.

1 INTRODUCTION

Durant leur cursus de DUT GEII, les étudiants appréhendent la programmation, en utilisant le langage C, dans le module Informatique du 1er semestre. Nous connaissons tous la difficulté actuelle à motiver et mobiliser nos étudiants dans leur apprentissage. Connaissant le goût des générations actuelles pour les jeux vidéo, il nous a semblé naturel d'assoir leurs connaissances des fondements de la programmation en C en leur proposant de développer une version simplifiée de Pong (cf. Figure 1), le premier jeu vidéo proposé par la société Atari au début des années 70. Dans cet objectif, 2 TPs de 3 heures leurs sont proposés dans le cadre du module Informatique du 1er semestre. Ces TPs font appel à la librairie *Simple DirectMedia Layer* (SDL) [2, 3] qui permet le développement de jeux vidéo en 2 dimensions. Ils complètent une série de neuf TDs de 2 heures et 4 TPs de 3 heures. Il nous a semblé intéressant que ces deux derniers TPs soient ludiques, tout en mettant en pratique les notions fondamentales du langage C enseignées jusque là (boucles, tableaux, fonctions, pointeurs, structures, etc …).
Dans la continuité de ces TPs plusieurs projets tuteurés du semestre 2 ont permis le développement de plusieurs jeux vidéo depuis 3 ans.

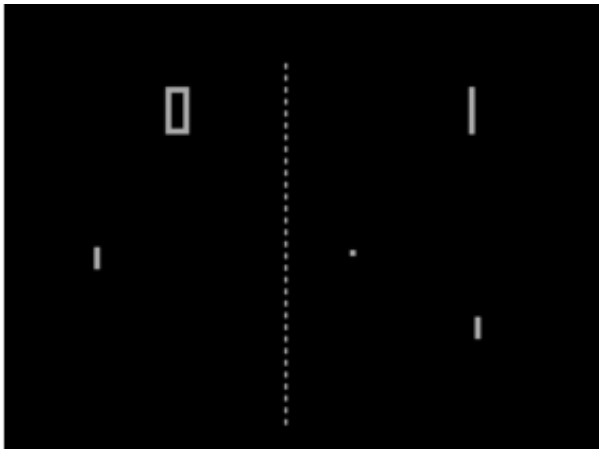

fig 1 : Pong original [1]

2 REGLES DU JEU

Voici les règles telles qu'elles sont présentées aux étudiants dans le fascicule de TP :

- La partie oppose deux adversaires (joueurs) : le camp bleu (à droite de l'écran) et le camp rouge (à gauche de l'écran) (cf. Figure 2).
- Initialement, la balle est au centre de la table et les raquettes, en l'absence d'interaction des joueurs, sont au centre verticalement, et horizontalement, à une distance égale à 1/4 de la largeur d'écran de leur propre ligne de fond de court.
- L'appui sur la barre d'espace permet de commencer le jeu.
- La raquette bleue peut être déplacée de la façon suivante :
  - L'appui sur la touche ↑ déplace la raquette verticalement vers le haut tant que la touche est enfoncée.
  - L'appui sur la touche ↓ déplace la raquette verticalement vers le bas tant que la touche est enfoncée.
  - Evidemment la raquette ne peut sortir des limites de la table.
- La raquette rouge peut être déplacée de la façon suivante :
  - L'appui sur la touche 'd' déplace la raquette verticalement vers le haut tant que la touche est enfoncée.
  - L'appui sur la touche 'c' déplace la raquette verticalement vers le bas tant que la touche est enfoncée.
  - Evidemment la raquette ne peut sortir des limites de la table.
- Les règles de déplacement vertical de la balle sont les suivantes :
  - La balle se déplace verticalement dans un sens donné (vers le haut ou vers le bas) jusqu'à ce qu'elle rencontre le bord du terrain (limite horizontale haute, ou limite horizontale basse).
  - Après la collision avec un bord horizontal du terrain, la balle rebondit avec un angle de 45° par rapport à la

verticale et change de sens vertical de déplacement.

- Les règles de déplacement horizontal de la balle sont les suivantes :
  - La balle se déplace horizontalement dans un sens donné (vers la droite ou vers la gauche) jusqu'à ce qu'elle rencontre :
    - Une raquette. Elle rebondie alors dessus avec un angle de 45° par rapport à l'horizontal en gardant le même sens vertical de déplacement (cf. Figure 2).
    - La ligne de fond de court. L'échange est alors terminé.
- Les déplacements horizontaux et verticaux sont évidemment simultanés.
- L'appui sur la touche q ou le fait de cliquer sur la croix de la fenêtre permettent de quitter définitivement le jeu.
- L'appui sur la touche h (help) permet d'afficher un écran d'aide résumant le rôle des différentes touches. L'appui sur la touche escape permet de l'effacer.
- Le score des camps Rouge et Bleu sont affichés dans une zone dédiée à cet effet sous le terrain de jeu. A chacune des extrémités de cette zone un rectangle de couleur montre le joueur ayant le service.
- Au service la balle part aléatoirement vers le haut ou le bas du terrain en direction de l'adversaire.
- Le service change tous les points. Le premier arrivé à 5 points gagne la partie.

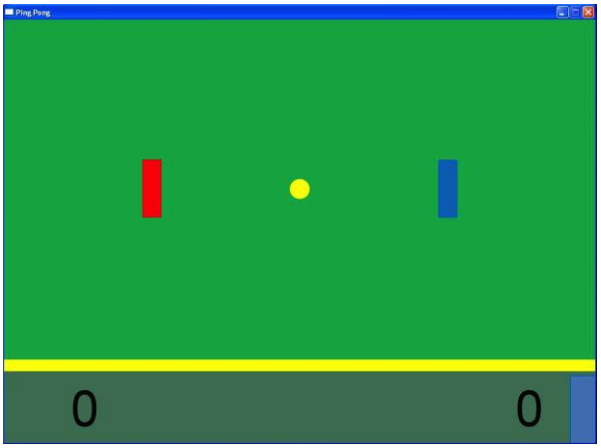

fig 2 : Ecran graphique demandé

3 OBJECTIFS VISEES

Malgré ce contexte ludique, outre l'utilisation des concepts de base de la programmation en langage C, compte tenu du volume du code à développer, les objectifs de ce TP sont ambitieux et multiples :

- Sensibilisation à la structuration des programmes,
- Compilation séparée, fichiers d'entête
- Maîtrise des structures de base du langage,
- Développement de fonction multi-paramètres avec renvoi de données,
- Utilisation des notions de pointeurs et structure,
- Gestion d'un affichage graphique,
- Interaction asynchrone avec un utilisateur, gestion d'événements,
- Notion d'objet graphique virtuel,
- Interaction virtuelle d'objets graphiques,
- Gestion du mouvement d'objets graphiques.

4 BIBLIOTHEQUE SDL

La bibliothèque *Simple DirectMedia Layer* (SDL) est une bibliothèque logicielle dont l'API est utilisée pour créer des applications multimédias en deux dimensions pouvant comprendre du son. Elle est donc adaptée aux besoins de la programmation d'un jeu vidéo.

Un exemple de programme SDL est présenté dans la figure 3. Ce code permet d'effectuer les actions suivantes :

- Initialisation de fenêtre :
  - Création d'un pointeur sur une surface (SDL_Surface), configuration du mode vidéo et nommage de la fenêtre,
- Affichage d'une image statique :
  - Création d'un pointeur sur une surface,
  - Chargement d'une image (SDL_LoadBMP),
  - Copie de l'image dans la fenêtre, appelée un *blit* (SDL_BlitSurface), aux coordonnées voulues (structure SDL_Rect),
  - Mise à jour de la fenêtre (SDL_Flip),
- Déplacement (mouvement) d'une image : même principe que précédemment avec l'utilisation d'une boucle for pour effectuer les copies de l'image, avec des coordonnées modifiées, et la mise à jour du contenu graphique de la fenêtre. Il est à noter que chaque image est affichée de manière superposée, c'est pourquoi il faut afficher à chaque itération l'image de fond puis l'image qui s'est déplacée.

Le tableau 1 présente les principales fonctions SDL utilisées.

Outre l'affichage d'images, il est également possible avec la bibliothèque SDL de gérer des événements i.e. l'utilisateur peut interagir avec l'affichage graphique par le biais d'actions sur une touche du clavier, un bouton de la souris ou en déplaçant cette dernière. Différentes fonctions sont à disposition pour caractériser un événement détecté.
La figure 4 présente le code permettant de gérer la gestion de l'appui sur la croix d'une fenêtre et l'appui sur

la touche ‘q’ (remarque : les codes claviers des touches étant gérés pour des claviers QWERTY, l’appui sur ‘q’ est détecté en identifiant l’appui sur ‘a’).

Enfin, il est possible d’écrire du texte en SDL, pour cela il faut *Blitter* (afficher une image sur une autre) une image représentant le texte voulu. Cela se fait avec la bibliothèque SDL_TTF.h. Un exemple est donné dans la figure 5.

```
int main(int argc, char *argv[])
{
    // Creation d'une fenêtre
    SDL_Surface *ecran = NULL;
    SDL_Init (SDL_INIT_VIDEO) ;
    ecran = SDL_SetVideoMode (640, 480, 32,
    SDL_HWSURFACE | SDL_DOUBLEBUF) ;
    SDL_WM_SetCaption (« Nom de la fenêtre », NULL) ;

    // Chargement d'une image de fond
    SDL_Surface *fond = NULL ;
    fond = SDL_LoadBMP ( « image_fond.bmp ») ;
    SDL_Rect position_fond ;
    position_fond.x = 0 ;
    position_fond.y = 0 ;
    // Blitter l'image
    SDL_BlitSurface (fond, NULL, ecran, &position_fond) ;

    // Mettre à jour la fenêtre
    SDL_Flip ( ecran ) ;

    // Chargement d'une image qui traverse le fond
    SDL_Surface *balle = NULL ;
    balle = SDL_LoadBMP (« image_balle.bmp ») ;
    SDL_Rect position_balle ;
    position_balle.y = 240 ;

    for(int i=0 ; i<640 ; i+=5){
        // Blitter la première image
        SDL_BlitSurface (fond, NULL, ecran, &position_fond) ;

        // Blitter la seconde image par-dessus
        position_balle.x = i ;
        SDL_BlitSurface (balle, NULL, ecran,
        &position_balle) ;

        // Mettre à jour la fenêtre
        SDL_Flip (ecran ) ;

        // Effectuer une petite attente pour « ralentir » la
        boucle
        SDL_Delay(10) ;
    }
    SDL_Quit () ;
}
```

fig 3 : Exemple de code SDL (1)

```
while (SDL_PollEvent(&event)) {
    switch (event.type) {
        case SDL_QUIT:
            // Action
            break;
        case SDL_KEYDOWN :
            switch (event.key.keysym.sym) {
                case ‘a’ :
                    //Action
                    break;
            }
    }
}
```

fig 4 : Exemple de code SDL (2)

```
// Initialisation
TTF_Init() ;

// Choix de la couleur du texte
SDL_Color couleur_texte_noire = {0, 0, 0} ; // format RGB

// Chargement de la police
TTF_Font *police = TTF_OpenFont(« arial.ttf », 90) ;

// Création de l'image correspondant à un texte
SDL_Surface * image = TTF_RenderText_Blended(police,
« Texte à afficher », couleur_texte_noire) ;

//Libération mémoire
TTF_CloseFont(police) ;
TTF_Quit() ;
```

fig 5 : Exemple de code SDL (3)

NOM DE LA FONTION	RÔLE
SDL_BlitSurface	Affichage rapide (en mémoire) de la surface spécifiée dans la surface de destination
SDL_Delay	Attente : nombre de millisecondes spécifié en argument
SDL_Flip	Affichage à l'écran de la superposition d'images créées en mémoire
SDL_FreeSurface	Libération des ressources utilisées par une surface SDL préalablement allouée
SDL_LoadBMP	Chargement d'un fichier Bitmap Windows dans une surface SDL
SDL_Quit	Libération des ressources utilisées
SDL_SetVideoMode	Configuration du mode vidéo (hauteur, largeur, nombre de bits)
SDL_WM_SetCaption	Nommage de la fenêtre

Tableau 1: Principales fonctions SDL utilisées

5 EXERCICES

Les étudiants doivent récupérer un répertoire qui leur est fourni. Dans ce répertoire se trouvent les images à utiliser et un projet Visual C++ déjà créé.
Durant les deux séances de TPs, les étudiants sont guidés par les items suivants :

1) Créer une fenêtre de jeu nommée « Jeu de Ping-Pong »,
2) Afficher dans la fenêtre l’image de fond de la partie dénommée « Bitmap/Fond.bmp »,
3) Afficher dans la fenêtre la balle au milieu du terrain,
4) Faire se déplacer la balle horizontalement entre les deux bords extrêmes du terrain,
5) Faire se déplacer la balle verticalement,
6) Faire se déplacer la balle en diagonale, et rebondir sur les 4 bords,
7) Gérer la fermeture de la fenêtre en détectant un clic de la souris sur la croix de la fenêtre,

8) Gérer la fermeture de la fenêtre en détectant l'appui sur la touche 'q',
9) Faire se déplacer la balle uniquement après l'appui sur la barre d'espace,
10) Afficher les raquettes à leur position d'origine respectives,
11) Gérer le déplacement de la raquette rouge avec les touches ↑ et ↓, gérer également la gestion des bords haut et bas : la raquette reste « coincée » quand elle atteint un des deux bords
12) Gérer de même le déplacement de la raquette bleue avec les touches c et d,
13) Encapsuler le déplacement de la basse dans une fonction,
14) Encapsuler le déplacement des raquettes dans une fonction.

Le but suivant est de modifier le déplacement de la balle pour prendre en compte les rebonds sur les raquettes. Elle doit rebondir en haut, en bas, et sur les raquettes, et revenir immobile à sa position d'origine quand elle atteint un des bords du terrain.

Il faut commencer par modifier le prototype de la fonction effectuant le déplacement de la balle. En effet, pour gérer la collision avec les raquettes, cette fonction doit également prendre en compte la position de chaque raquette, ce qui s'avère difficile en utilisant uniquement des entiers. Afin de faciliter la gestion et l'affichage des images, on propose d'utiliser une structure regroupant toutes les informations pertinentes correspondant à l'image :

```
typedef struct Figure_info
{
        char * name;
        int hauteur;
        int largeur;
        int x_init;
        int y_init;
  int x;
  int y;
  int delta_x;
  int delta_y;
};
```

15) Modification de la balle, des raquettes et des fonctions,
16) Modification de la fonction gérant le déplacement de la balle pour qu'elle gère les collisions avec les raquettes,
17) Gestion avancée du jeu :
    a) Au démarrage du programme, une image s'affiche pour indiquer quel joueur est au service, le choix du joueur au service est aléatoire au lancement du jeu, puis change à chaque point,
    b) A chaque début de point, la balle part en haut ou en bas de façon aléatoire,
    c) L'affichage des points est affiché en bas de l'écran,
    d) Lorsqu'un joueur atteint 5 points, un affichage indique quel joueur a gagné et le programme se termine.
18) Affichage de l'aide lors de l'appui sur la touche 'h', et disparition de l'aide lors de l'appui sur la touche escape.

6 RETOUR D'EXPERIENCE

Ces travaux pratiques sont dispensés pour la quatrième année pendant le module Informatique du semestre 1 de DUT GEII.

En trois ans, nous avons pu constater que l'engouement des étudiants pour un thème plus ludique est dominé par la difficulté de la tâche à accomplir. Au fur et à mesure des années, l'équipe enseignante améliore sa façon de présenter le problème, pour décomposer au maximum les étapes à effectuer et aider les étudiants à mener à bien ce projet. Seuls quelques binômes arrivent au bout des tâches à effectuer. Ce sont généralement des étudiants passionnés par la programmation et prenant plaisir à préparer le programme chez eux. Les étudiants qui, malheureusement, ne passent que les 6 heures de TP à travailler sur le projet, arrivent généralement à effectuer les 14 premiers points : faire rebondir la balle sur les 4 côtés du terrain et faire bouger les raquettes. Les étapes suivantes nécessitant l'utilisation de fonctions puis d'une structure sont difficilement réalisables par les étudiants en difficulté avec ces notions.

7 PROJETS TUTEURES

Le thème ludique « programmation d'un jeu vidéo » est ensuite proposé durant le semestre 2. Ce sont bien sûr des étudiants ayant apprécié et réussi les TPs précédents qui choisissent ce type de sujet. Différents jeux sont proposés chaque année et certains étudiants proposent même d'eux même un jeu qu'ils ont envie de programmer (comme Mario par exemple, qui a toujours beaucoup de succès).

Durant ces projets, un cahier des charges est établi en début de projet par les étudiants en accord avec leur encadrant. A partir du thème principal du jeu, les étudiants énoncent les fonctionnalités qu'ils s'engagent à accomplir, et celles qu'ils pourraient aborder si le temps le permet. Il arrive régulièrement que les étudiants proposent des fonctionnalités plus avancées (et non vues en TP) que ce que l'encadrant envisageait : utilisation de la musique, gestion des mouvements de la souris … Les étudiants sont laissés le plus possible en autonomie. Le rôle de l'encadrant se limite à les aider en cas de blocage, et à leur faire des propositions pour améliorer la lisibilité et la structuration de leur code. Le but est de

leur faire comprendre qu'un programme qui fonctionne n'est pas suffisant si le code correspondant n'est pas bien écrit. Les encadrants incitent donc fortement à l'utilisation de fonctions et de structures.

Depuis trois ans, plusieurs jeux vidéo ont vu le jour :

- Casse Briques (cf. Figure 6),
- Space Invaders (cf. Figure 7),
- Démineur (cf. Figure 8),
- Candy Crush (cf. Figure 9),
- Mario (cf. Figure 10).

Un jeu de carte a également été développé, avec la possibilité de jouer en réseau joueur contre joueur, ce qui a nécessité un gros travail, travail qui n'avait pas été demandé dans le cahier des charges initial.

8 CONCLUSION

La programmation de jeux vidéo est une approche ludique qui permet aux enseignants de motiver les étudiants depuis plusieurs décennies [4]. Nous avons présenté 2 séances de Travaux Pratiques qui permettent, dès le premier semestre, de programmer dans le langage C le jeu de Pong. Ces séances ont pour but de mettre à profit les notions apprises précédemment dans le cadre d'une activité plus ludique. Mettant en œuvre une programmation que l'on peut qualifier d'"avancée", nombre d'étudiants ne verront plus les jeux vidéo avec le même regard.
Les étudiants les plus passionnés par la programmation peuvent ensuite, s'ils le souhaitent, effectuer un projet tuteuré dans cette lignée, durent le deuxième semestre. Depuis 3 ans, chaque année, entre un et trois nouveaux jeux vidéo voient le jour.

Bibliographie

[1] https://fr.wikipedia.org/wiki/Pong
[2] http://www.libsdl.org
[3] https://fr.wikipedia.org/wiki/Simple_DirectMedia_Layer
[4] T.-T. Dang-Ngoc and T. Liu, « Mario, Warcraft ou Pokemon ? La programmation de jeux vidéo comme motivation dans l'apprentissage des sciences informatiques », Colloque d'enseignement des technologies et des sciences de l'information et des systèmes – CETCIS'14.

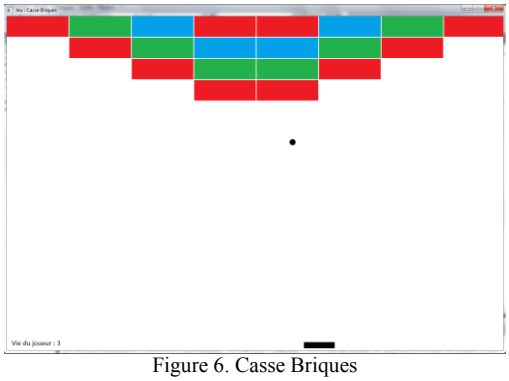

Figure 6. Casse Briques

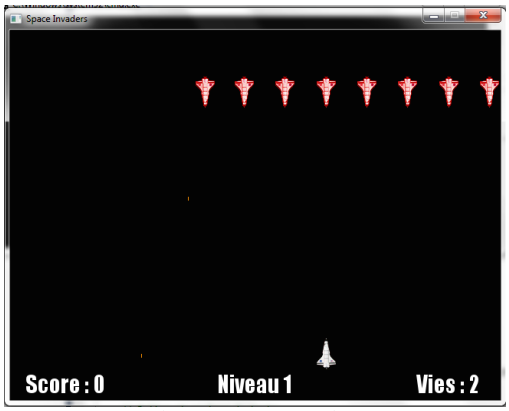

Figure 7. Space Invaders

Figure 8. Démineur

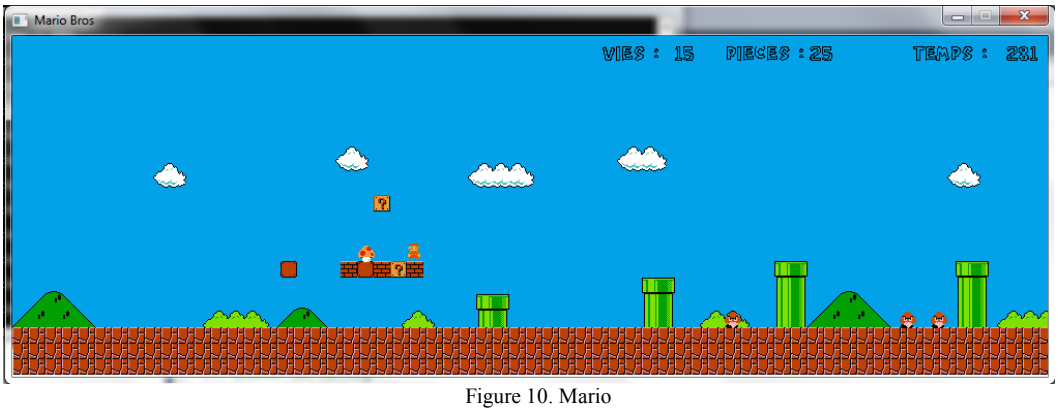

Figure 10. Mario