Jeu sérieux, un oxymore?

Florence Quinche[1]

Créer des jeux vidéo pour apprendre la logique algorithmique

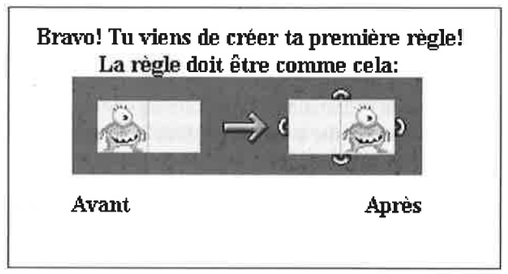
Depuis quelques années sont apparus nombre de logiciels de création de jeux vidéo destinés aux enfants et adolescents. Il est possible de créer différents types de jeux, de la simulation en 2D, 3D, des fictions interactives. Mais quels peuvent être les apports pédagogiques de ces logiciels en classe?

Création de simulations et de jeux en 2D

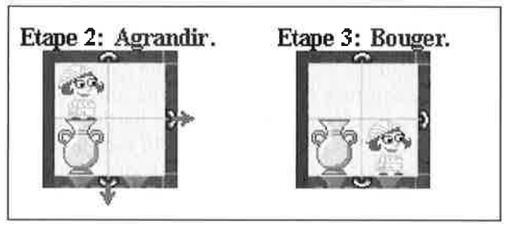
Des logiciels ont été produits directement dans un objectif pédagogique et proposent un ensemble de documents pédagogiques et d'exemples de réalisations d'élèves (par exemple *Stagecast creator* www.stagecast.com). Le principe de ce type de logiciel est de programmer des objets ou figures (personnages), afin de générer certains comportements. L'acquisition des principes de programmation étant très progressive, cela permet à des élèves très jeunes de découvrir les bases du raisonnement algorithmique.

Apprentissage des règles

Avec ce type de logiciel, les mouvements d'un personnage ou d'un objet sont programmés en posant des règles (algorithmes). Un mouvement est une transformation où l'on passe d'un état à un autre. Comme il s'agit de jeux proposant une interface numérique vidéo, ce que l'on programme, ce sont des images. Pour donner l'impression d'une modification d'état du personnage, il s'agira de faire se succéder des images différentes. Par exemple, l'image d'un ours «statique» est remplacée par celle d'un ours qui marche, court, saute, etc. Pour que ce passage se fasse automatiquement, on pose une règle qui définit l'état initial et l'état final visé. Mais pour activer une règle, elle doit être déclenchée par quelque chose. Dans ce logiciel, les personnages se déplacent sur des plateformes, franchissent des obstacles, récoltent des objets. Les événements (obstacle, escaliers, trou dans le sol, autre personnage, etc.) que rencontre l'avatar dans ses déplacements activent des règles. Par exemple, face à un espace vide, l'avatar, s'il a été programmé avec la règle ci-dessous, devra avancer d'une case:



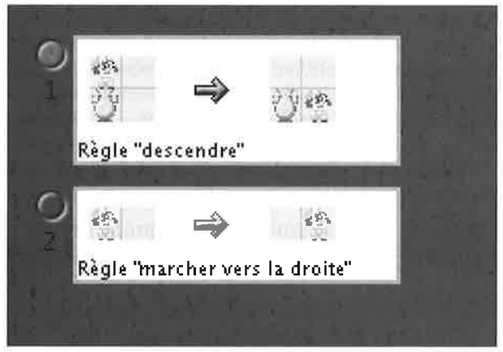
En fonction des objets rencontrés dans le jeu, on programme des règles différentes. Par exemple, lorsque l'avatar se trouve face à une marche d'escalier, il devra pouvoir sauter pour monter. Lorsqu'il se trouvera face à un trou dans le sol, l'élève programmera une chute à l'étage inférieur, etc. Ce sont des changements d'état de l'avatar qui sont ainsi anticipés (règles d'action). Voici une règle pour faire descendre le personnage. Seuls deux états sont nécessaires (état initial et état final):



La structure de pensée qui correspond à ce genre de règles est la pose de conditionnelles du type: «Si l'événement Y se produit, alors il se passe Y.»
Pour tester la compréhension des règles par l'élève, on peut passer d'un mode de représentation à un autre (changement de registre), par exemple en traduisant le langage par pictogrammes du programme en langue naturelle, et inversement. A noter que l'utilisation d'un langage algébrique n'est pas nécessaire à ce stade. Il s'agit donc d'une introduction à l'algorithmique qui peut être préalable à l'apprentissage de l'algèbre.
Plusieurs niveaux de règles peuvent s'articuler, car certaines règles concernent spécifiquement l'application d'autres règles. On parle alors de «métarègles» (par exemple une règle qui demanderait l'application en boucle d'une autre règle ou suite de règles). On peut également utiliser ces métarègles pour articuler des suites de règles dans un système. Le programme ne se construit plus alors simplement en accumulant successivement des règles singulières, mais en composant des groupes de règles, ou des suites d'actions (cela est possible dans le programme *Scratch*).

Cohérence des systèmes de règles

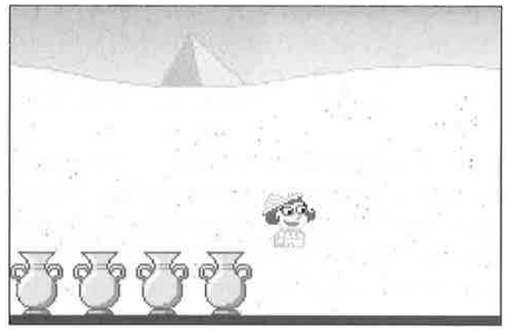
L'élève apprend ensuite à programmer des règles pour différentes situations, à imaginer de multiples états possibles. Mais l'ensemble des règles posées doit être cohérent (pour un même personnage). Les avatars (l'ours, le tigre, etc.) peuvent cependant avoir des comportements distincts (d'autres règles de fonctionnement). Chaque personnage aura donc son propre système de règles, ses possibilités et limites: chacun peut réagir différemment à l'environnement ou réaliser des actions distinctes.
L'ordre des règles choisies est également important, car l'avatar les réalise toujours dans le même ordre (il les examine une à une et exécute la première qui est applicable dans l'environnement donné, puis la seconde, etc.). Si l'on change cet ordre, cela peut modifier le comportement de l'avatar. Il faut donc trouver un ordre qui corresponde aux actions que devra réaliser le personnage dans le jeu. L'élève apprend ainsi à construire une syntaxe logique (un ensemble logique de règles). Dans cette phase de conception, les règles sont construites comme des hypothèses, car elles demandent à être vérifiées. Il s'agit donc de tester dans le jeu le comportement de l'avatar avec les règles qui lui ont été assignées. La notion de règle est ainsi à penser directement en relation avec un environnement (et non de façon purement abstraite).
Voici un exemple d'une liste de deux règles posées dans l'ordre suivant: le personnage descend (si un espace libre en dessous de la prochaine case se présente), puis marche vers la droite (si l'espace est libre).



Vérifier ses hypothèses

La différence entre cette programmation et un problème mathématique classique, c'est qu'il n'y a pas de bonne solution, tout dépend de ce que l'on veut faire accomplir à ses avatars. Plus que de trouver déductivement «la bonne solution», il s'agira de construire un ensemble cohérent de règles, de manière à ce que le programme génère les interactions voulues dans une situation donnée. On peut dire que le test des hypothèses (règles posées) est *empirique*, bien qu'il ait lieu sur une interface numérique souvent appelée à tort «virtuelle». C'est en observant comment se comporte le personnage programmé que l'élève vérifie ses hypothèses. Il y a donc une combinaison de pensée hypothétique (création de règles) et analytique (analyse du fonctionnement de l'algorithme).

En reprenant le petit système de règles précédent, si l'élève se trompe et met en premier la règle «marcher à droite» et seulement ensuite la règle «descendre d'une case», son avatar va marcher dans les airs (et ne jamais descendre), car tant que la règle 1 est applicable, la règle 2 n'est pas exécutée (donc le personnage ne redescend pas sur le sol):



Lors de la phase d'observation, on constate souvent que le fonctionnement de l'avatar ne correspond pas à ce qui était attendu (vérification expérimentale). L'élève va s'interroger sur ce dysfonctionnement (phase d'analyse des erreurs dans le programme). Il est alors nécessaire de reformuler les règles, voire de changer leur ordre d'application (production d'un nouveau modèle). Avec ce type de logiciels, il est aussi possible de réaliser des simulations de phénomènes naturels complexes, car on peut symboliser et représenter de nombreux types d'interactions.

Conclusion

Cette activité permet de travailler l'apprentissage de procédures automatisées et, par ce moyen, la modélisation (PER, MSN 15, 25, 35). Mais on exerce également les bases de la méthode expérimentale: poser des conjectures, les vérifier, observer le résultat, analyser, puis remodeler les hypothèses. L'élève découvre également de façon ludique les principes de la cybernétique: étude de la situation initiale (données de départ), réflexion sur le traitement de ces données (règles), puis analyse des données de sortie. •

[1] Philosophe de la communication, Prof. form., UER MT, HEPV.

15 logiciels de création de jeux: www.netpublic.fr/2012/03/creer-des-jeux-video-15-logiciels-gratuits.

Bibliographie

Eduscol, (2009) *«Algorithmique», Mathématiques, Ressources pour la classe de seconde*, Ministère éduc. nat., France (avec le logiciel *Scratch*)
J. P. Gee (2003), *What video games have to teach us about learning and literacy*, New York, Palgrave Macmillan
F. Quinche (2013), *Apprendre avec les jeux vidéo*, éd. Fondation educa.ch, Educaguide, en ligne
http://guides.educa.ch/fr
F. Quinche (2010), «Les avatars du virtuel, entre incarnation, métamorphoses et symbolisme», *in* P. Weber & J. Delseaux (dirs.), *De l'espace virtuel, du corps en présence*, PUN, 2010, 145-153