

blender art

MAGAZINE

L'apprentissage facile de Blender

Réaliser un personnage en Low Poly

Retouche de Materiaux

Créer une animation Cartoon

Apprendre le BGE

Blender et le jeu Réseau

Etude de cas BRE

COUVERTURE - Cristian Mihaescu 'Espoir pour un dinner'

Cartoon & GE

EDITEURGaurav Nawani gaurav@blenderart.org**REDACTEUR EN CHEF**Sandra Gilbert sandra@blenderart.org**SITE INTERNET**Nam Pham nam@blenderart.org**DESIGNER**

Gaurav, Sandra, Alex

RELECTEURS

Kevin Braun
Phillip Ryals
Bruce Westfall
Joshua Leung
Lynda Schemansky
Eric Pranausk
Noah Summers
Joshua Scotton
Mark Warren
Wade Bick
Patrick O'Donnell
Brian C. Treacy
Scott Hill
Henriel Veldtmann

AUTEURS

Brian Cordell Hynds
Brian Treacy
Husam Ibrahim
Igor Krianovskij
ititrx
John Buresh
Mal Duffin
Nanmo
Rogério Perdiz
Tobias Dahl Nielsen

COUVERTURE

Cristian Mihaescu 'Espoir pour un dinner'

SOMMAIRE

2

Modéliser & Rigger une araignée Cartoon**8****Créer un modèle Low poly à partir d'un modèle High Poly****15****Retouche de Matériaux en utilisant Nodes et Vertex Color****19****Création d'une animation cartoon avec Blender****21****Apprendre les bases du Blender Game Engine****23****Blender et le Jeu en réseau****43****Making of - Orion Tear****50****Making of - Teenage Duck****60****Making of - Monkey Game Project****65****Etude de cas - Le moteur temps réel de Blender****68****Interview - 'M. Blender des 'Services Web'****73**



Sandra Gilbert
Rédacteur en chef

Même après des années d'utilisation de Blender, la large palette des possibilités créatives de Blender me stupéfie toujours. Vous pouvez créer tout ce que vous imaginez, depuis des dessins animés aux jeux. Dans ce numéro, c'est cette même diversité que nous allons explorer.

En grandissant, je suis toujours restée fascinée par les dessins animés et encore aujourd'hui, j'aime en regarder. Particulièrement avec toutes les avancées technologiques qui ont ouvert la voie à toujours plus d'exploits techniques pour les possibilités d'animation.

Les programmes de modélisation de 3D ont changé l'aspect des dessins animés mais les concepts de base développés par "les maîtres du dessin animé" sont restés les mêmes. Les programmes 3D, y compris notre Blender bien aimé, nous offrent la liberté de créer des dessins animés de n'importe quel style nous venant à l'esprit. Une chose qui, bien que possible, était beaucoup plus difficile avec les techniques traditionnelles. Par rapport à l'animation "traditionnel", les dessins animés deviennent un moyen populaire d'expression artistique..

Comme l'animation 3D devient plus populaire, la limite entre le dessin animé et l'animation se rétrécit et devient de plus en plus floue, au point que les deux termes en deviennent interchangeable. En fait, avec le nombre croissant de films d'animation entièrement produits en 3D, la définition de dessin animé s'étend aussi rapidement que les esprits imaginatifs des animateurs les créent. La diminution du coût de logiciels de modélisation très chers et les programmes libres comme Blender, permet à tout le monde d'essayer la création de ses propres dessins animés et animations. Ouvrant le champ des dessins animés/animations à une nouvelle vague de jeunes amoureux de dessins animés qui deviendront par la suite des créateurs.

Se fondant sur la popularité croissante des dessins animés, les jeux deviennent rapidement une force artistique à part entière. De nos jours, vous pouvez trouver des jeux basés sur beaucoup de dessins animés, animations et films po-

pulaires aussi bien que des concepts originaux. En fait "le jeu" est devenu une grande industrie avec des occasions croissantes pour ceux qui sont habiles dans la modélisation, le texturing et même l'animation de personnages. Beaucoup de sociétés de jeux ont sérieusement besoin d'artistes 3D et je prends le pari que le besoin continuera à progresser tant que les avancées technologiques et la possibilité pour des jeux plus intenses graphiquement grandira. La création de jeux utilise beaucoup les mêmes compétences et talents que l'animation. La connaissance de la modélisation, du texturing et de l'animation de personnage sont une grande aide que vous créez votre premier chef-d'œuvre de jeu ou votre centième.

Un intérêt particulier pour nous est le fait que Blender peut être utilisé soit pour créer votre maquette de jeu afin de l'exporter dans un moteur de jeu séparé soit pour créer des jeux directement dans Blender lui-même. Le Moteur de jeu de Blender a vu un grand nombre d'améliorations au cours des années, menant un segment en croissance de notre communauté à apprendre à développer des maquettes de jeu, des démonstrations et des jeux complets. En même temps que la communauté grandit, les solutions sont trouvées et de nouvelles techniques sont créées. Les façons d'utiliser les logic bricks de Blender sont plus efficaces, aussi bien avec que sans codage Python supplémentaire, ce qui a mené au nombre croissant de jeux étant créés et disponibles dans les [forums GE](#). La variété de jeux créés est aussi imaginative que les artistes qui ont consacré leur temps au GE.

Donc si vous êtes animateur, créateur de jeu bourgeonnant ou simplement curieux des possibilités de Blender que vous n'avez pas encore apprises, nous vous les ferons découvrir. Nous avons rassemblé de grands articles pour augmenter votre créativité et la rendre opérationnelle.

Bon Blend!

sandra@blenderart.org

IZZY CONNAÎT : L'Importance de la Composition

4



*Une bonne composition
peut faire une image
ennuyeuse se démarquer,
vous faire asseoir et
dire 'wouaou'!*

Après avoir récemment lut un [post du blog de Virgilio](#) sur la composition et le cadre, j'ai réalisé que même si je connaissais la "règle des tiers" depuis certain temps, ce n'était plus quelque chose à laquelle je pensais réellement. Ce qui est dommage, car une bonne composition peut faire une image ennuyeuse se démarquer, vous faire asseoir et dire 'wouaou'!

Dans une certaine mesure, un grand nombre d'artistes utilisent la règle des tiers inconsciemment quand ils créent leurs images, règle qui curieusement n'est pas aussi étrange que vous pourriez le croire. Il y a certaines proportions et placements de points de focalisation auquel l'homme répond instinctivement. Ainsi, bon nombre d'entre nous placent tout simplement les objets instinctivement et les déplacent au hasard jusqu'à ce que le résultat semble correct.

Mais, comme pour beaucoup de choses dans la vie, prendre un moment pour considérer l'impact de la composition sur votre image ou sur les plans de votre animation, permet d'aboutir à une image plus dynamique qui engage pleinement le spectateur.

Réalisant que la composition n'était plus le fruit d'une décision consciente lorsque je m'asseyais pour créer une image, j'ai ressorti et étudié quelques-une des images que j'ai fait depuis quelques années. Sans surprise, les images dont j'étais vraiment content suivaient la "règle des tiers" et celles qui semblaient n'avoir jamais assez d'heures de travail souffraient d'une mauvaise composition et d'une absence totale de point de focal.

J'ai trouvé l'exercice très enrichissant et j'encourage quiconque cherchant à améliorer ses images et animations à essayer par lui-même. Quand vous avez fini de regarder vos propres images, essayez avec d'autres. Souvenez-vous de regarder tous types d'images, aussi bien les bonnes que les mauvaises.

Si vous n'êtes pas familiarisé avec la "règle des tiers", je vous suggère de jeter un coup d'œil aux deux articles suivants. Ils ont tous les deux de très bonnes explications et ont des exemples visuels qui sont d'excellents entraînements à la composition.

[Learning Composition with the New BlueSky Trailer](#)
www.methart.com/tutorials/composition.html

Ressources du Game Engine, jeux et démos

Le [forum du Game Engine](#) possède un certain nombre d'excellents tutoriels et démos expliquant des fonctionnalités variées du Game Engine, aussi bien que des méthodes pour mettre en place différents types d'éléments tels que des menus, des IA ordinateur/ennemis, des tirs de projectiles, etc. Le forum contient un mine de renseignements qu'il serait bien trop long de lister ici. Je vais donc me contenter d'en mentionner quelques-uns que j'ai vu récemment et qui ont attiré mon attention comme étant des utilisations extrêmement utiles et ingénieuses des capacités du GE.

- [Modèle d'éditeur de niveau](#)
- [Bots autonomes sans Python](#)
- [IA Simplifiée](#)
- [Modèle de jeu d'aventure à la 3ème personne](#)
- [Tutoriel de sensibilité aux tirs de rayons](#)
- [Modèle de FPS avec le BGE](#)

La documentation officielle peut être trouvée sur le [Wiki de Blender](#) :

- [Documentation du Game Engine](#)
- [BSOD Introduction au Game Engine](#)

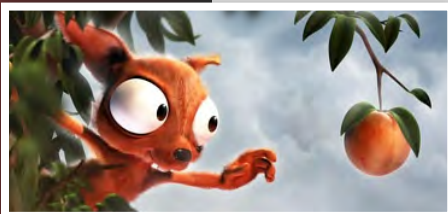


*Une bonne composition
peut faire une image
ennuyeuse se démarquer,
vous faire asseoir et
dire 'wouaou'!*

Échantillons de jeux et démos actuellement en développement

Plus d'informations et d'état d'avancement sur ces jeux/démos peuvent être obtenus en cliquant sur les liens listés.

- [1st MBGP - Kart Racer!](#) : en fait un nouveau projet d'apprentissage qui va durer un mois, période au bout de laquelle un nouveau projet sera lancé.
- [Project Echo](#)
- [Mon premier jeu RPG](#) (par Rusty246)
- [Duk! Duk!](#)
- [GunFlash](#)
- [My Tetris Game](#)
- [Shoot'n'score](#)



Le Project Peach

Le Projet Peach avance gaiement, apportant de nouvelles améliorations et fonctionnalités excitantes à Blender. Mes préférées, évidemment, sont les nouvelles capacités de cheveux/fourrure. Si vous n'avez pas encore vu ces

derniers effets de cheveux et de fourrure, vous ratez quelque chose.

Pour bénéficier des mêmes fonctionnalités que l'équipe de Peach, vous aurez besoin de vérifier le dépôt SVN de développement, puisque tous les composants requis ont été ajoutés depuis la version 2.45

Les nouvelles fonctionnalités actuellement présentes dans le dépôt SVN

Vous pouvez déjà trouver quelques projets avec le dépôt SVN. Les rapports de Bug et les Réactions sont vivement attendus. Vous pouvez en savoir plus [ici](#).

- Amélioration du Skinning
- Fonctions de Rendu
- Modificateur Mesh Deform
- Pôle Cible pour IK
- Réflexion/Réfraction brillante
- Ombres Douces Raytracées
- Échantillonnage QMC
- Visualisation du weight paint et Multi Modificateur
- Render Baking et Normal Mapping

- Particules et fils

Le Project Apricot



Les membres de l'équipe du Projet Apricot se rencontreront la 2ème semaine de janvier lors d'un long week-end pour les préparatifs de production, mais le vrai démarrage aura lieu la première semaine de février. Alors seulement, de vrais décisions seront prises sur le concept, le design du jeu et d'autres points. Cependant, nous savons qu'il sera dérivé des personnages poilus et dingues du projet Peach évoluant dans une forêt.

Comme pour les précédents films libres, une campagne de pré-vente aura lieu pour les DVD afin que les communautés Blender et Crystal Space puissent supporter le projet. A nouveau, le but ambitieux est d'atteindre 1000 DVD vendus avant le 1er février.

Le DVD devrait contenir le jeu complet, tout le contenu sous licence Creative Commons Attribute, les blogs vidéos, des exemples clairs et/ou tutoriaux pour pouvoir créer des extensions au jeu, des niveaux, des modifications de personnages et plus encore. Tous les noms des pré-acheteurs du jeu Apricot seront cités dans les génériques d'intro et de fin.

Et n'oubliez pas : bien que nous visons à la création d'un contenu de jeu convaincant, l'objectif réel d'Apricot est d'améliorer le pipeline open source pour la création professionnelle de jeu. Cela, concentré autour de Blender comme modèleur et outil d'animation, Crystal Space comme plateforme et moteur 3D et un peu de magie du script Python pour lier tout ensemble.

Guide PDF de modélisation de précision.

Vous vous souvenez peut-être de Robert Burke de l'excellent tutoriel (Réaliser des objets avec précision) qu'il a écrit dans le BlenderArt Magazine n°11 (Mécanique). Nous sommes heureux d'annoncer et de promouvoir le reste de son travail sur la modélisation de précision dans Blender. C'est une excellente référence que nous encourageons tous les utilisateurs de Blender à consulter.



Descriptif extrait du site web de Robert Burke

Le Guide de la Modélisation de Précision avec Blender est un manuel de 151 pages sur les outils de modélisation et des techniques de design. Illustré de 600 images, il donne une introduction claire et précise sur l'utilisation de Blender. En parcourant le guide, vous serez initié aux principaux outils de modélisation de Blender mais aussi à certains moins connus. Les outils sont présentés dans un exemple de conception réel avec les explications du choix de leur usage et non pas simplement une démonstration de leur utilisation.

Ne soyez pas repoussé par le nom, le guide est utile à tous ceux qui veulent une introduction claire aux outils de modélisation de Blender.

Stats: 151 Pages ; 613 Images; 4.85 Mb

Télécharger le Guide de la Modélisation de Précision [ici](#).



Introduction

Les outils de modélisation de Blender sont excellents pour beaucoup de tâches, particulièrement pour faire des formes organiques. Faire un personnage de dessin animé simple et drôle avec Blender est aussi facile qu'une promenade dans un parc (une fois que vous savez comment faire, évidemment). J'essayerai de le prouver dans ce court tutoriel. Ainsi, vous aurez d'abord besoin d'une idée. Si vous n'avez pas d'idée neuve, utilisez-en une vieille, elle est probablement déjà assez mûre pour être recyclée avec de nouvelles dimensions. Tel fut le cas pour moi... En réalité, vous pourriez même sauter cette partie; avec Blender vous n'avez même pas besoin d'avoir une idée. Jouez avec les outils et leurs options est d'habitude plus que suffisant pour commencer!

Niveau Débutant à Intermédiaire

Ma vieille illustration d'araignée était un point de départ parfait pour un film 3D court et idiot, et pour essayer quelques nouveaux outils de Blender dans des secteurs tels que : les lumières, les scènes, la peinture, les matériaux, le rigging, les particules, les shape keys, le NLA, les noeuds, le montage vidéo, le son, les codecs, etc..

Donc j'ai eu besoin d'un bon modèle, mais un qui soit aussi assez simple pour réaliser des temps de rendu convenables/courts. Vous savez, des temps de rendus courts => film plus long dans un temps plus court.

Il y a beaucoup de bons tutoriels de modélisation, y compris dans les précédents numéros de BlenderArt (<http://www.blenderart.org/>), consultez-les donc. La plupart des outils de modélisation sont déjà matures, donc ils n'ont pas beaucoup changé.

Ici je vous montrerai mes étapes clés du processus de modélisation et de rigging:



par Igor Krianovskij

Astuces de Modélisation

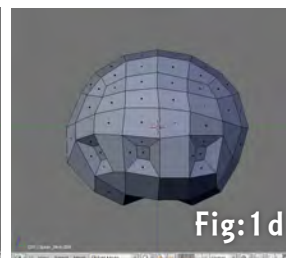
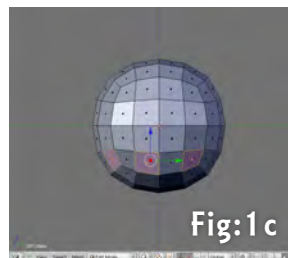
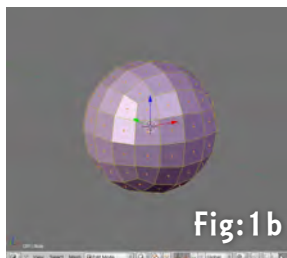
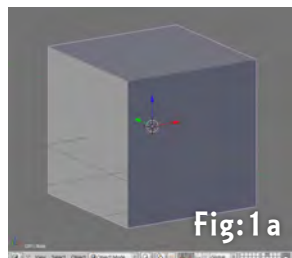
Les outils de modélisation de base sont : Déplacement et mouvement [G] (Grab), l'Échelle [S](Scale) et la rotation [R]. Vous pouvez choisir des pièces de maillages joints avec [L] dans le Mode édition (la modélisation est faite dans ce mode la plupart du temps). Une fois sélectionné, vous pouvez les séparer de nouveau avec [P]. Pour faire une face (aussi connu sous le nom de polygone) sélectionnez les arêtes ou les sommets puis touchez [F]. Avec [ALT+S] vous pouvez déplacer les faces le long de leurs normales.

par Igor Kriapovskij

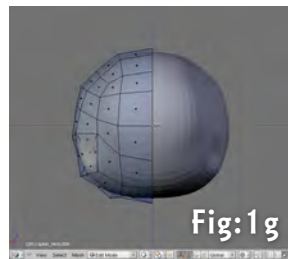
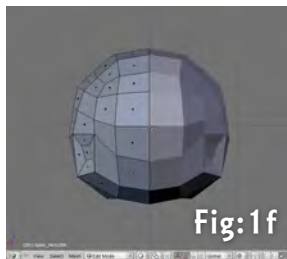
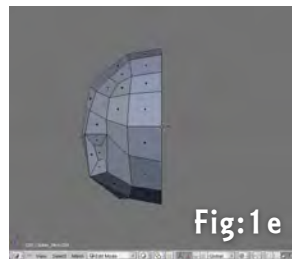
LE CORPS

Il est préférable de commencer votre modèle au centre de l'espace 3D, particulièrement si vous utiliserez les Modificateurs de Miroir, les Armatures, etc. Pour commencer tapez [SHIFT+C] pour déplacer le curseur 3D au centre et "NumPad1" pour la vue de Face (pour que l'axe "X" aille de gauche à droite).

Ajoutez un cube (en mode d'Objet : [ESPACE], Add > Mesh > Cube) et ajoutez-y un Modificateur Subsurf ([F9] pour les boutons d'Édition, puis Add Modifier > Subsurf), mettez le Level à 2 et appuyez sur le bouton Apply dans le panneau Modifiers (image 1.h) (ou [ALT+C] dans le mode objet). Ajustez les sommets pour obtenir une forme comme celle-ci.



Sélectionnez 3 faces et extrudez les vers l'intérieur avec [E] afin de faire une place pour les 'connexions' des parties supérieures des jambes avec le corps. Vous pouvez supprimer un côté du mesh du corps et ajouter un modificateur Mirror, vous n'aurez alors à ajuster qu'une moitié du corps; d'où le fait qu'il est d'habitude bon de construire des choses symétriques en premier. Ok, voilà la forme du corps de l'araignée. Facile, hein! ? Nous ajouterons les poils ou la fourrure plus tard...



LA TÊTE

Regardez le croquis pour voir de quelle taille doit être la tête pour s'adapter avec le style de l'illustration. Vous pouvez être flexibles ici, en la faisant plus grande ou plus petite. Parfois vous vous obtenez de meilleurs résultats quand vous ne suivez pas exactement le croquis.

Il y a plusieurs façons de construire la tête pour un corps. J'ai utilisé la topologie du corps comme point de départ pour la tête. Ainsi, j'ai dupliqué les faces du corps (4x3) en Mode édition avec [SHIFT+D]. Ces 12 nouvelles faces faisaient alors toujours partie du mesh du corps, donc je les ai séparés en utilisant [P].



Fig:2a

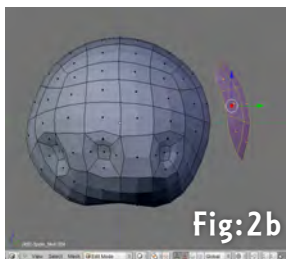


Fig:2b

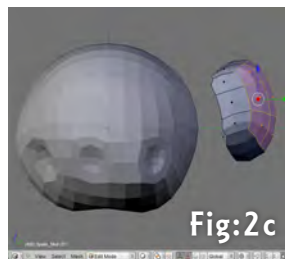


Fig:2c

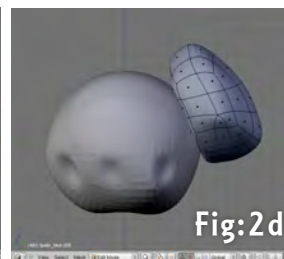


Fig:2d

Vous pouvez faire la tête de la même manière que le corps au-dessus... à partir d'un cube. Mais en faisant un double de la partie du mesh du corps déjà existant vous obtenez l'alignement précis des sommets utiles pour assembler plus rapidement les parties du corps (moins d'ajustement, résultat plus rapide!).

Ainsi, après quelques extrusions [E] et certains ajustements de quelques sommets, j'ai obtenu ceci:

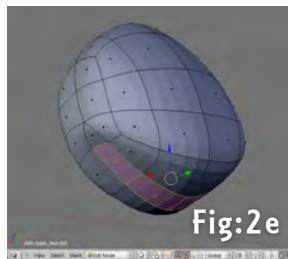


Fig:2e

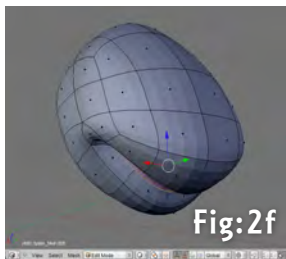


Fig:2f

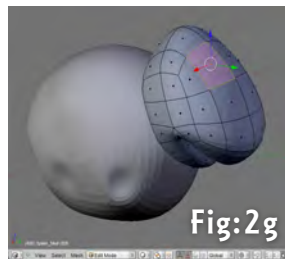


Fig:2g

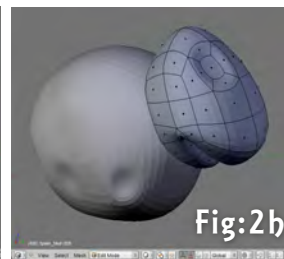


Fig:2h

Astuces de Modélisation

Dans Blender, [X] est un raccourci clavier extrêmement important, qui est d'habitude utilisé pour supprimer les oeuvres d'art laide et maladroite! Intéressant comment Blender est efficace avec quelques raccourcis clavier, hein!?

LES JAMBES

A nouveau, avec le même principe que la tête, j'ai sélectionné une face (ce qui était suffisant pour cette partie) où le «joint» de la jambe du milieu sera placé. Je l'ai dupliqué avec [SHIFT+D] et séparée avec [P]. Après quelques extrusions [E] et sommets ajustés, j'ai obtenu ceci :

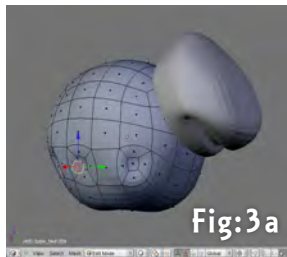


Fig:3a

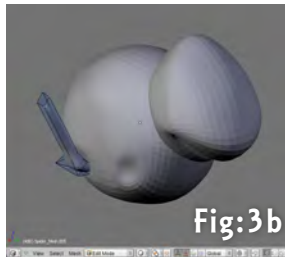


Fig:3b

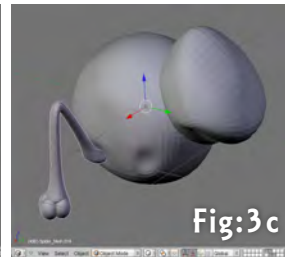


Fig:3c

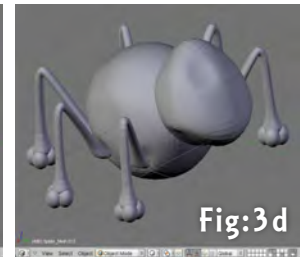


Fig:3d

Au bout de la jambe j'ai ajouté quatre sphères (voir la note 1 en bas de l'article) et joint tous les meshes de la jambe ensemble avec [CTRL+J]. Je pourrais aussi poser la jambe dans une position de repos différente, mais pour une étape ultérieure il est important que les genoux soient au moins un peu pliés afin d'effectuer convenablement les calculs d'une Armature IK plus tard. Finalement j'ai dupliqué les cinq jambes restantes dans les positions prévues (de nouveau, avec [SHIFT+D]).

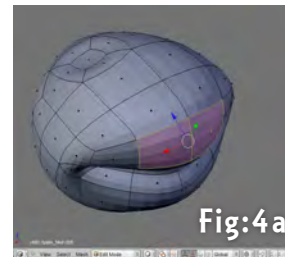


Fig:4a

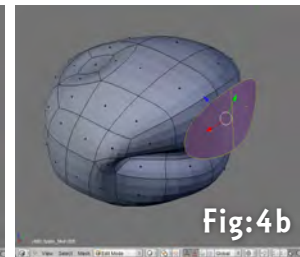


Fig:4b

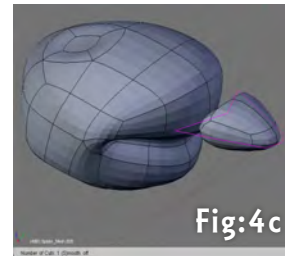


Fig:4c

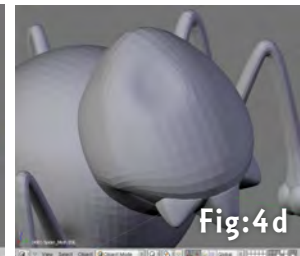


Fig:4d

LES YEUX ET LES DENTS

Les dents de l'araignée sont super simple... enfin, comme le reste, hein! ? Ok, j'admet, j'ai procédé de la même manière encore une fois! [SHIFT+D], [P], [E], ajustage... Mais cette fois j'ai ajouté un Loop Cut supplémentaire avec [K] (image 4c.) pour rendre la topologie des dents de l'araignée plus captivante!

Astuces de Modélisation

Utilisez le Point Médian et les pivots de Curseur 3D avec les outils Snap tools [SHIFT+S] pour le positionnement du centre de l'axe de paupière. Face Loops et Edge Loops peuvent être choisis avec [ALT+SHIFT+RMB] et [ALT+CTRL+RMB]. Outils sournois!

Nouveau Système de Particule : Vérifiez le BlenderWiki ActionBook pour les versions mises à jour!
http://wiki.blender.org/index.php/Blender_3D: ActionBook

par Igor Kriapovskij

Je pourrais faire les yeux plus simples, mais il y a une petite astuce pour la 'lentille de l'iris' d'un l'oeil. À savoir, vous obtiendrez de meilleurs réflexions de lumière si vous rendez la lentille plus sphérique ou convexe. Voici comment je l'ai fait : Ajoutez une sphère en utilisant [ESPACE], ensuite Add > Mesh > Uvsphere (avec 8 Segments et 8 Rings). Sélectionnez les 16 faces du dessus (2 anneaux) et séparez-les avec [P]. Rendez la lentille plus convexe en la déplaçant [G] (Grab) et en modifiant l'échelle [S] (Scale).

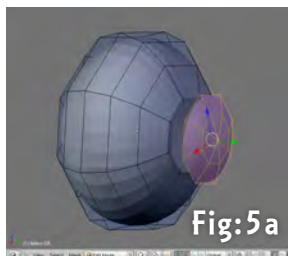


Fig:5a

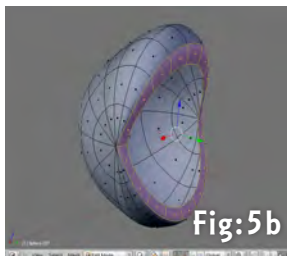


Fig:5b

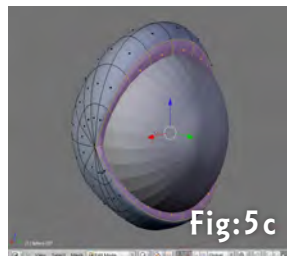


Fig:5c



Fig:5d

La paupière est faite avec une UV-Sphère supplémentaire (12 Segments et 8 Rings), tournée de 90 degrés par rapport au globe oculaire. Modifiez ensuite l'échelle, supprimez une rangée de segments, extrudez une Edge Loop vers le centre de la paupière (image 5 b-c). Pour l'ouverture/fermeture de la paupière j'ai utilisé les ShapeKeys, mais je ne rentrerai pas dans le détail des Shape-Key cette fois. Vous pouvez regarder le [fichier](#) joint pour les détails.

Poils, Fourrure et Autres Affreuse Excroissances!

La modélisation de l'araignée à cette étape est terminée pour l'essentiel. Le reste concerne les couleurs et les affreux poils, bien sûr! Ce qui est encore une fois très facile. Les couleurs ont été ajustées dans le Menu de Shading [F5] et les Émetteurs de Particule utilisent des meshes dupliques à partir du maillage déjà existant de l'araignée.

Le Système de Particules des versions de développement actuelles de Blender (janvier 2008) est si avancé que le fichier fait dans la version 2.45 de Blender est déjà périmé. Aussi, les particules de ce fichier ne seront pas compatibles avec les futures versions de Blender! Vérifiez le fichier joint pour connaître la configuration des Particules (Utilisez [F7] pour les boutons Physics et le panneau Particle system).

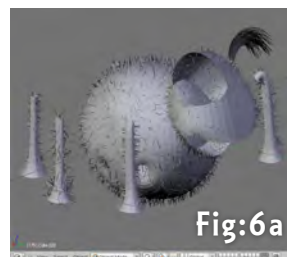


Fig:6a



Fig:6b



Fig:6c



Fig:6d

ASTUCES DE RIGGING

Si vous voulez éviter les comportements d'Armature imprévisible, commencer à construire l'armature au centre [SHIFT+C]!

Si vous voulez gagner du temps dans le nommage des bones, nommer seulement les bones importants '(Root_*, Handle_*, Tag_*) et laisser Blender nommer le reste des bones automatiquement! Faites seulement attention d'ajouter *.R ou *.L sur la fin de certains noms osseux (par exemple : Handle_Leg_01.L). Blender le reconnaîtra comme une chaîne osseuse sur le côté gauche. C'est aussi utile pour l'édition des bones en Miroir!

par Igor
Kriapovskij

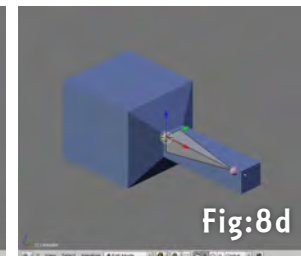
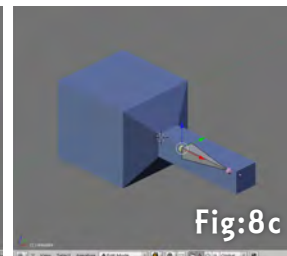
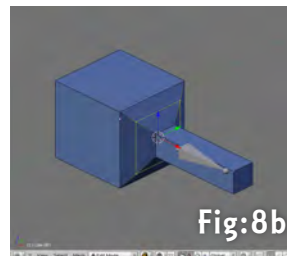
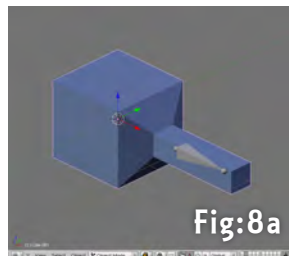
Rigging, Huile et Sueur!

La phase suivante vers l'animation fut le rigging. Cette armature contient l'utilisation de base (version 1 - voir la note 2 à la fin de l'article) des contraintes IK, mais pour une animation je n'ai pas eu besoin d'un rigging plus complexe. Ainsi, j'ai ajouté le premier bone au centre (Mode Objet: [ESPACE], ensuite Add > Armature). Je nomme d'habitude ce premier os Root_'quelquechose', dans ce cas Root_Spider.



Après, j'ai dupliqué ce bone dans le Mode d'Édition, j'ai extrudé [E] les extrémités des bones deux fois et les ai positionnés de nouveau avec l'outil Snap [SHIFT+S].

Voici un exemple de configuration pour mettre un tibia dans la position exacte, ce qui est très difficile à réaliser manuellement, particulièrement en travaillant avec des formes organiques et des angles exotiques:



8a) Add Bone ; **8b)** Régler le pivot sur Median Point; Sélectionnez les arêtes; [SHIFT+S], puis choisir Cursor -> Selection; **8c)** Sélectionnez l'extrémité du bone dans le Mode Edition; **8d)** [SHIFT+S], puis choisir Selection -> Cursor

ASTUCES DE RIGGINS

Vous pouvez commuter entre les Modes Pose et Edit avec [TAB] et entre les modes Pose et Objet avec [CTRL+TAB].

N'oubliez pas de désactiver les Enveloppes dans le Modificateur d'Armature si vous utilisez les Vertex Groups!

ASTUCE PRINCIPALE FINALE : Une animation par jour garde Kaito au loin!

par Igor Kriapovskij

Ok, retournons au rigging de l'araignée. Après avoir extrudé et placé tous les bones nécessaires pour une jambe, j'ai commencé la configuration d'une chaîne logique IK. Les bones avec contrainte IK solvers sont colorés en jaune. Chaque jambes de l'araignée a un IK solver (configuré à ChainLen:2) que j'ai ajouté dans le mode Pose (l'image 9b.). Après avoir dupliqué les jambes des côtés gauches/droites [SHIFT+D], j'ai ajouté un bone pour la tête et un bone pour la bouche/mâchoire. Il y a aussi une poignée facultative pour déplacer toutes les jambes simultanément!

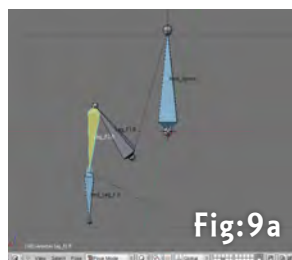


Fig:9a

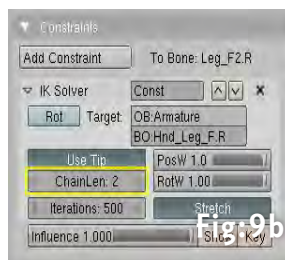


Fig:9b



Fig:9c

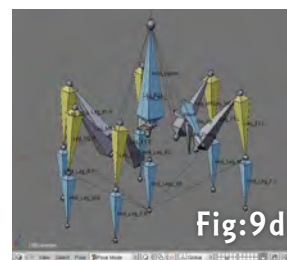


Fig:9d

Finalement tout ce qui restait à faire fut d'attacher le mesh à l'Armature. Ce fut très simple. J'ai ajouté un Modificateur d'Armature au mesh et j'ai assigné des Vertex Groups à chaque bone.

Téléchargez le .blend de l'araignée [ici](#).

Araignée dans le parc. - Notes

1. Ajouter des sphères de cette façon n'est pas optimale si vous construisez le modèle pour un jeu. Il ne doit pas y avoir de faces cachées dans ce cas! Ce modèle a été prévu pour une animation, donc ces sphères 'denses' peuvent être utilisées comme des émetteurs de particules si nécessaire.
2. Le système d'Armature dans Blender a été construit par étapes. La version 1 peut faire les FK, IK et quelques autres contraintes de base. La version 2 peut faire des choses plus avancées(consulter le Mancandy's rig réalisé par Bassam Kurdali comme exemple). La version 3 est la dernière et meilleure version du système d'armatures utilisé par le Blender Institute pour les personnages du prochain "Open Movie"! •

ATELIER 3D : Créer un modèle Low poly à partir d'un modèle High Poly

15



NDT: Le retopo rend
obsolète cette
technique depuis
la version 2.48 !

par ititrx

Introduction

Ceci est un court tutoriel qui décrit une technique pour créer un modèle Low Poly adapté aux jeux vidéo, en se basant sur sa version High Poly. Il existe plusieurs technique pour celà (Box modeling, combing shapes ?!, etc), donc c'en est une parmi d'autre. Utilisez votre technique de modélisation préférée pour ce qui va suivre.

Niveau: Débutant à intermédiaire

Ouvrez Blender et supprimez la caméra par défaut et le cube. Sélectionnez le calque 2 et importez un modèle High Poly de votre choix. Divisez votre écran en 3 vue, 2 vue 3D dans la partie supérieure et le panneau des boutons au dessous. Paramétrez les 2 vues 3D en vue de face et vue de coté.

Sélectionnez les calques 1 et 2. Vous modéliserez votre Low Poly sur le calque 1 grâce au High Poly sur le calque 2. Vous pouvez faire vos face aussi large que vous le souhaitez, et aussi proche du High Poly que vous voulez. Si vous voulez vérifier votre modélisation, cliquez simplement sur le calque 1. Une autre solution serait de laisser le modèle sur le calque 1 et de le cacher/montrez pour l'utiliser.

ETAPE 1: Placez le curseur 3D sur l'avant du High Poly en vue de face et vérifiez la profondeur sur la vue de coté, comme illustré sur la figure 1. Dans la vue de face, insérez un Plane.

ETAPE 2: Avec le Plane toujours sélectionné en mode Édition, ajoutez un modifier Mirror, comme sur la Fig 2, pour vous simplifier la modélisation.

ETAPE 3: Déplacez le Plane sur le coté du plan de symétrie, comme sur la Fig 3.



ATELIER 3D : Créer un modèle Low poly à partir d'un modèle High Poly

16

ETAPE 4: Dans la vue de coté, tournez le Plane pour correspondre à l'angle du modèle. Fig 4.

ETAPE 5: Sélectionnez le coté droit du plane et extrudez le vers la droite. Fig 5.

ETAPE 6: Avec le coté droit toujours sélectionné, reculez le en suivant le coté du High Poly. Fig 6. Vous voyez comment le point du bas suit le modèle ?

ETAPE 7: Extrudez le même coté encore vers la droite, et déplacez le autour du High Poly. Fig 7.

ETAPE 8: La Fig 8 montre le résultat après avoir extrudé les deux arrêtes du haut et les avoir ramené en arrière jusqu'à ce qu'elles touchent le High Poly.

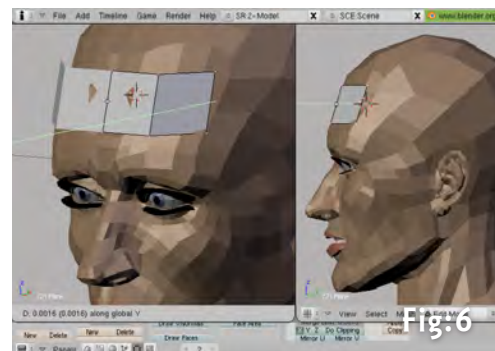


Fig:6

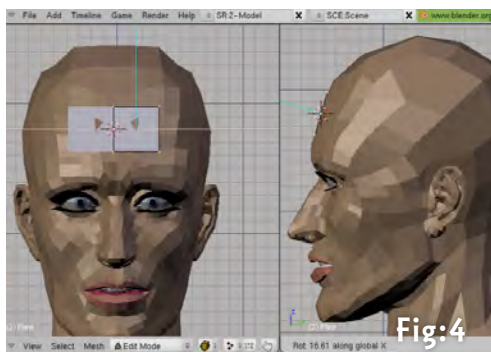


Fig:4



Fig:7

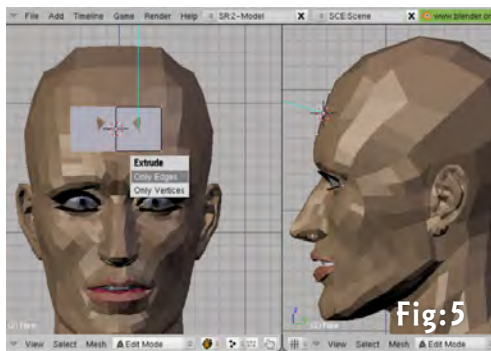


Fig:5



Fig:8

par ititrx

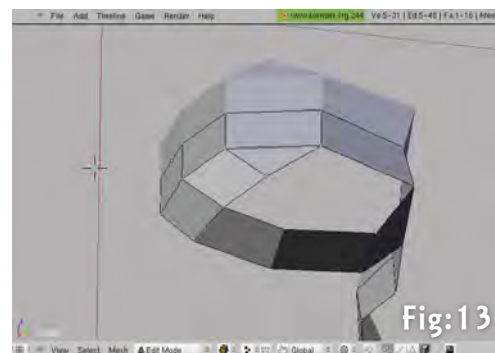
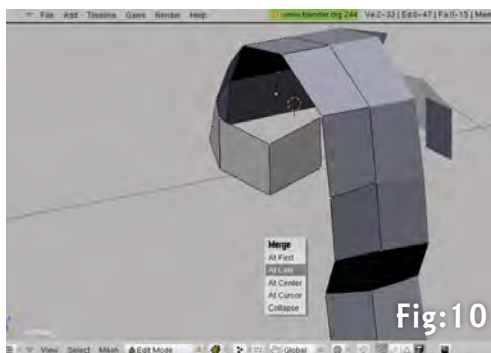
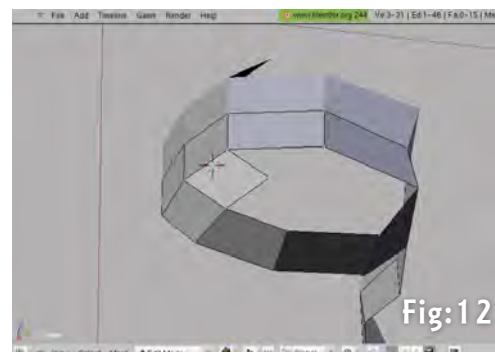
ATELIER 3D : Créer un modèle Low poly à partir d'un modèle High Poly

17

ETAPE 9: Continuez de cette manière pour couvrir la tête du High Poly. Fig 9.

ETAPE 10: Les Fig 10 et 11 montrent la sélection du calque 1 uniquement pour mieux voir la modélisation. Deux point ont été sélectionnés et fusionnés.

ETAPE 11: Les Fig 12 et 13 montrent la création de face à partir de points sélectionnés pour couvrir/fermer les ouvertures entre les faces.



par ititrx

ATELIER 3D : Créer un modèle Low poly à partir d'un modèle High Poly

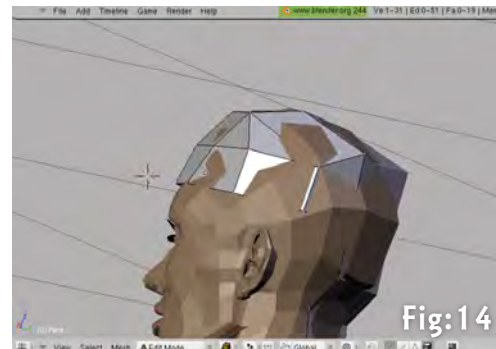
18

ETAPE 12: En sélectionnant à nouveau les calques 1 et 2, vous verrez qu'il peut y avoir des zones laide où les points s'éloignent trop du High Poly. Fig 14.

ETAPE 13: Avec quelques modifications, les cornes disparaissent. Fig 15.

ETAPE 14: Sur la Fig 16, les cotés du bas ont été extrudés.

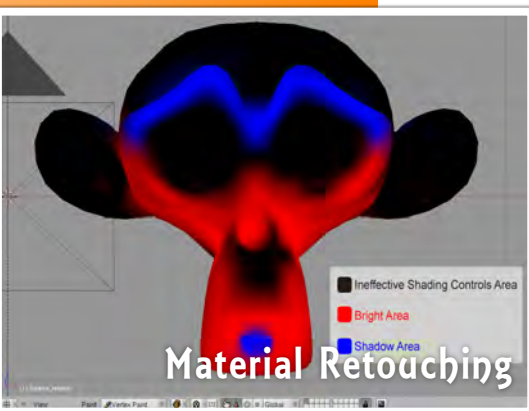
Continuez de cette manière pour couvrir le modèle entier. Des changements ont été nécessaires dans la topologie pour réduire le nombre de polygone. Avec de la pratique, vous serez capable de faire le modèle Low Poly en environ 30 minutes. Si vous avez des problèmes avec des zones de votre modèle, vous aurez peut être à subdiviser des faces pour aider à les lisser. •



par ititrx

ATELIER 3D : Retouche de Matériaux en utilisant les Nœuds de Matériaux et Le Vertex Color

19



Introduction

La Retouche de matériaux est une méthode pour contrôler une zone ombrée en utilisant les Nœuds de Matériaux avec le Vertex Color. Cela est utile si vous n'avez pas obtenu l'effet désiré sur une minuscule zone d'ombre d'un modèle. Évidemment, vous pourriez refaire le modèle, mais cela prendrait du temps. Une autre approche serait d'utiliser le post processing, typiquement en employant un Nœud de Composition pour donner un effet 2D, mais cela crée une vue 3D partiellement incorrecte. Or vous pouvez rapidement ajuster l'ombrage du modèle

avec la Retouche de matériaux.

Voyons comment la Retouche de Matériaux fonctionne

Premièrement, en mode Vertex Paint, nous peindrons deux zones de contrôle des ombres, une en rouge et une en bleu. Peignez vos secteurs lumineux en rouge, vos secteurs d'ombre en bleu et les zones sans importance en noir.

Figure.1: Exemple de peinture Vertex Color.

Deuxièmement, créez des Nœuds à partir du Matériau original comme figuré dans l'image ci-contre.

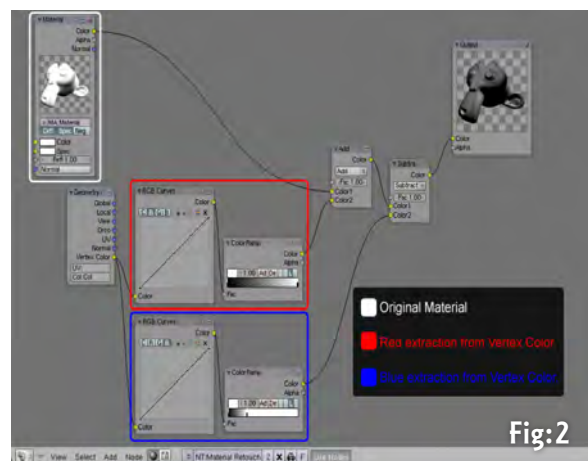
Figure.2: Nœud de Matériaux de la "Retouche de Matériaux".

Note: Pour extraire le Rouge ou le Bleu des Vertex Color utilisez un Nœud RGBCurve. Pour extraire le Rouge, mettez une valeur zéro aux canaux vert et bleu du nœud RGBCurve. Maintenant vous pouvez rendre votre scène et continuer à ajuster les couleurs jusqu'à ce que vous obteniez l'effet désiré.

Figure.3: Gauche: ombres normales de Suzanne. Droite : "Retouche concave" des ombres. L'éclairage est fourni par une lampe Sun.

Cette méthode est basée sur les couleurs car Blender 2.45 ne supporte pas le contrôle d'influence des ombres.

Notez que la Retouche Matériel affecte seulement les couleurs, pas les ombres Ray-tracées ou "Buffered".



par Nampio

ATELIER 3D : Retouche de Matériaux en utilisant les Nœuds de Matériaux et Le Vertex Color

20

Références:

Ombres Stylisées Contrôlables Localement (Anglais)

http://www.olm.co.jp/en/rd/2007/07/siggraph_2007_paper.html

Ombres Stylisées Contrôlables Localement (Japonais)

http://www.olm.co.jp/b/rd/2007/07/_-locally_controllable_stylized_shading.html

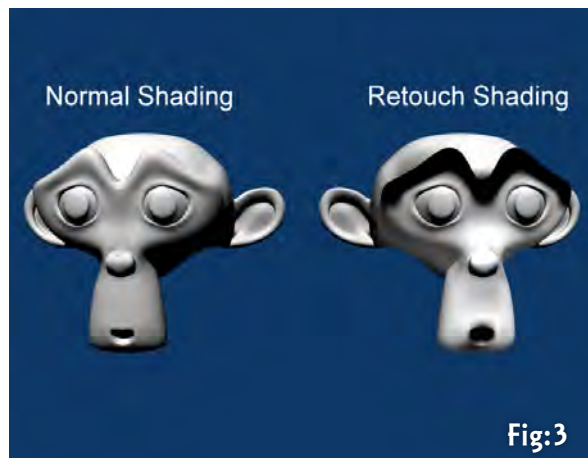
Test de retouche d'ombres Toon en 3D (Japonais, XSI)

<http://artifacts.sakura.ne.jp/sakanaya/2007/10/3d.htm>

Crédit:

Nanmo(Japon)

<http://bleble.s321.xrea.com/>



par Nanmo



Animation Cartoon

Introduction

Ce qui fait qu'un cartoon est très amusant à lire c'est qu'ils sont vraiment 'à la puissance 20'. Ils sont pleins de personnages disproportionnés aux mouvements totalement irréels et c'est ce qui les rend intéressants. Une autre chose importante quand on crée des cartoons, c'est de les rendre faciles à lire et à regarder, avec des couleurs simples et claires et des ombres précises. Ce tutoriel vous apprendra comment créer une petite animation cartoon en utilisant ces règles. Bon cartooning !

Première Partie (Organisation du plan de travail et de l'animation)

Note: si vous voulez juste appliquer un effet cartoon à une scène que vous avez déjà créée, vous pouvez directement passer à la seconde partie.

La raison pour laquelle j'ai mis une partie "Organisation du plan de travail" au lieu de "modélisation" ou "armaturage" est que je voulais que cela soit simple. Nous allons donc créer une simple sphère, bien sûr les principes de l'animation de cette sphère peuvent être utilisés pour tout ce que vous animez. Commencez par supprimer le cube par défaut et sélectionnez la vue de dessus. Créez une Sphère-UV puis allez dans l'onglet "Edit" et cliquez sur "Set Smooth". Ensuite, sélectionnez la vue de face et mettez la sphère juste au-dessus de la ligne rouge. (l'axe des abscisses (x))

Nous allons utiliser la ligne rouge comme un repère, pour savoir où le sol se trouve quand nous sommes en vue de face. Créez un cube, réduisez-le beaucoup et positionnez-le au-dessus de la sphère. Le cube sera notre 'rig'. Sélectionnez la sphère, puis cliquez sur l'onglet "Object" et cliquez ensuite sur "Add Constraint", enfin sur "Stretch To". Écrivez "Cube" dans le champ "Target". Essayez de déplacer le cube.

La sphère devrait maintenant s'étirer vers le cube, tout en conservant son volume (si vous éloignez le cube au loin, la sphère

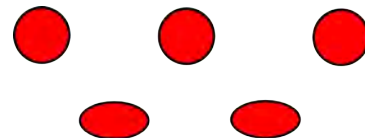
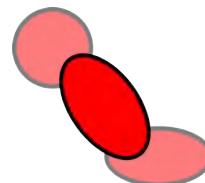
deviendra plus mince, etc.). Allez dans la vue de dessus et créez un plan. Déplacez-le de façon à le mettre juste en dessous de la sphère. Mettez-vous en vue caméra et redimensionnez-le jusqu'à atteindre les limites du champ. Divisez la vue 3D en haut et transformez la nouvelle fenêtre en fenêtre "timeline".

Sélectionnez la caméra et appuyez sur [Alt+R]. Effectuez une rotation de 90° sur la caméra sur l'axe des x et placez cette dernière juste en face de la sphère. Allez à la frame 0 et cliquez sur le bouton rouge en forme de cercle dans la fenêtre "timeline" (cela enregistrera les endroits où vous placez la balle). Sélectionnez la vue de la caméra puis sélectionnez en même temps la sphère et le cube et déplacez-les ensemble sur l'axe des x de manière à ce qu'ils soient en dehors du champ de la caméra.

Déplacez les ensemble un petit peu. Maintenant, nous sommes prêt pour commencer! Bougez la balle de haut en bas toutes les 10 frames, comme montré sur l'image (la première balle commence à la frame 0, la suivante frame 10, et ainsi de suite) et utilisez le cube pour 'écraser' la balle quand elle touche le sol en le rapprochant de la balle.

Souvenez-vous qu'il faut bouger la balle et le cube en même temps! Aux frames intermédiaires (5,15,25 et ainsi de suite), étirez la balle, de manière à ce que le bout pointe vers la pose clé précédente et l'autre vers la suivante, comme le montre l'image. L'écrasement - étirement est vraiment un principe de base ! Les objets qui sont en mouvement s'étirent tandis que ceux qui entre en collision avec quelque chose s'écrasent.

Note : Ne vous attendez pas à avoir des résultats satisfaisants du premier coup, il vous faudra un peu d'entraînement afin de perfectionner tout cela.



Seconde partie (matériau, lumière et configuration des nodes)

Premièrement, allez dans l'onglet Scene et cliquez sur le bouton 'Edge'. Ensuite, cliquez sur 'Render Layers' et sélectionnez 'Col' et 'Sha'. Ce sont les passes pour la couleur (Color) et les ombres (Shadow) (nous en aurons besoin pour la configuration des nodes). Deuxièmement, cliquez sur le bouton en dessous du mot 'Scene' et sélectionnez 'ADD NEW'. Cela va créer un nouveau RenderLayer. Ensuite, décochez les boutons 'Solid', 'Halo', 'Ztra' et 'Sky'. Nous aurons besoin de cela pour utiliser 'Edge' dans la configuration des nodes.

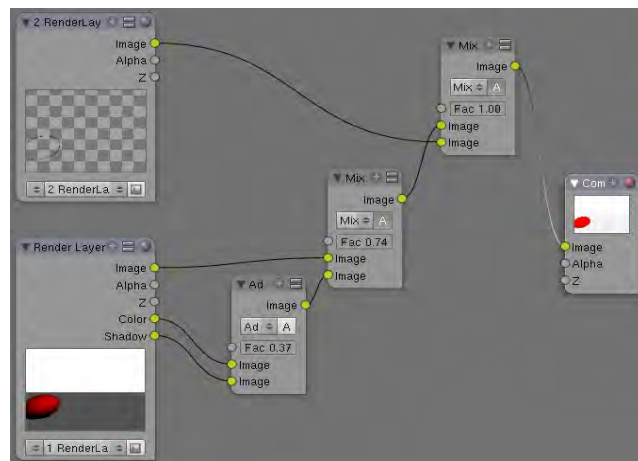


Materiaux: Les matériaux doivent simplement être de couleur claire et sans spécularité (si vous pouvez faire sans). J'ai choisi un rouge clair pour la sphère. Evitez aussi le 'Ray-Mirror' parce que ça ne va pas avec le style cartoon. Il est conseillé de donner un matériau complètement transparent au cube, et de ne pas le rendre 'traceable' ou spéculaire.

L'éclairage: L'éclairage peut se composer d'une seule lampe sun, 'brillant' en haut, à droite de l'image. NOTE : Ca, c'est seulement pour ce projet en particulier. Vous pouvez évidemment utiliser la configuration que vous voulez.

Les nodes: Comme je l'ai dit précédemment, nous allons avoir un RenderLayer séparé pour les contours toons (Edges). En fait, c'est un truc que j'ai trouvé pour résoudre un problème que j'ai rencontré avec les nodes qui ne montraient pas les contours à certains endroits de l'image (ceux en face du sol, pour être exact). Voici la configuration complète des nodes:

Cela consiste à utiliser deux RenderLayers (celui par défaut et celui que nous avons créé). Nous ajoutons les passes 'Color' et 'Shadow'



depuis le premier RenderLayer, créant ainsi l'ombrage cartoon ('Fac' devrait être aux alentours de 0.35). Ensuite, on 'mix' l'output à partir de là, avec l'image (avec alpha) pour faire apparaître le ciel ('Fac' aux alentours de 0.74). Enfin, on 'mix' l'output à partir de là avec l'image du second layer (avec alpha), créant ainsi le contour toon ('Fac' aux alentours de 1). Vous pouvez jouer avec le paramètre fac pour arriver à l'effet que vous souhaitez exactement. Cependant, j'ai trouvé que c'est ce qui marchait le mieux pour moi. Assurez-vous d'avoir connecté les différents nodes de la même manière que moi. Maintenant, tout ce que vous avez à faire est de sélectionner 'AVI Raw' (ou n'importe lequel des formats de film que vous aimez) dans 'Format', et cliquer sur 'Anim'. Voilà ! Votre animation cartoon est faite !

Vous pouvez voir une version plus longue de l'animation faite dans ce tutorial [here](#).

Si vous avez un problème ou un commentaire, vous pouvez m'envoyer un email: tobiasdn@gmail.com.



Introduction

Ce tutoriel va vous montrer comment créer un jeu en 3D simplement et à partir de zéro avec Blender. Après avoir terminé le didacticiel, les compétences acquises vous permettront de prolonger le jeu et ses niveaux. Pour faire ce didacticiel, une connaissance minimum de Blender est requise, mais pas pour le Game Engine interne à Blender. Seuls les aspects de Blender qui sont pertinents pour faire des jeux simples seront abordés.

Ce tutoriel devrait prendre environ 2 heures pour créer un jeu 3D à partir de zéro

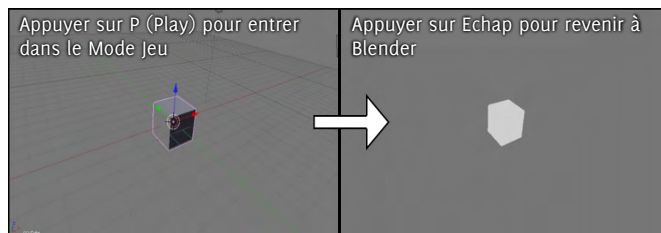
Niveaux: Débutant à Intermédiaire

Apprendre les bases d'utilisation du Game Engine de Blender

Vous pouvez lancer et quitter le Game Engine en appuyant sur P (pour le lancer) et sur Echap (pour le quitter).

Nous allons voir quelques éléments de Blender dans le GE (Game Engine). Commençons avec le raccourci clavier le plus important dans Blender : celui qui vous permet de démarrer le GE. Déplacez votre souris dans la scène 3D et appuyez sur P pour jouer (Play) le jeu. Bravo, vous venez juste de jouer à votre premier jeu dans blender !!! C'était facile, non ?

Comme nous n'avons pas demandé au GE de faire quoi que ce soit, il ne se passera rien dans la scène. Pressez Echap pour revenir à Blender.



Mise en place de la scène par défaut

Avant que nous commençons, réinitialisez la scène dans Blender pour revenir aux paramètres par défaut. Cela peut être fait avec une des trois méthodes ci-dessous...

Sélectionnez "Fichier (File)" >> menu "New", et cliquez sur l'option "Erase All".

Appuyez sur [Ctrl+X], et cliquez sur l'option "Erase All"

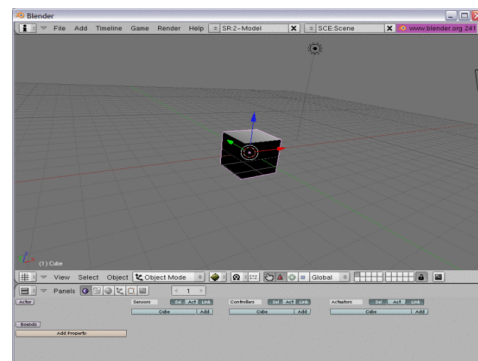
Appuyez sur [Ctrl+Q] pour quitter Blender (versions 2.42 et supérieures, appuyez juste sur [Q]). Ensuite redémarrez Blender manuellement.

Il est maintenant temps de voir comment vous pouvez utiliser Blender, en utilisant les informations que vous avez apprises dans la section précédente. Configurez la vue 3D dans Blender comme sur le screenshot. Pour faire cela, vous allez avoir besoin d'effectuer les opérations suivantes...

Effectuez une rotation de la scène en utilisant le bouton du milieu de la souris (BMS).

Changez la vue en mode perspective en cliquant sur "View" >> "Perspective" (ou appuyer sur [5] sur le clavier numérique).

Ajoutez un cube et une lampe (en supposant que vous les aviez supprimés précédemment et qu'ils ne font pas partie de la scène par défaut lorsque Blender charge un nouveau fichier).



Une séquence de touches très utile lorsque l'on travaille avec le GE

Une touche utile à retenir lorsque l'on travaille avec le GE de Blender est celle qui maximise la fenêtre 3D en cours.

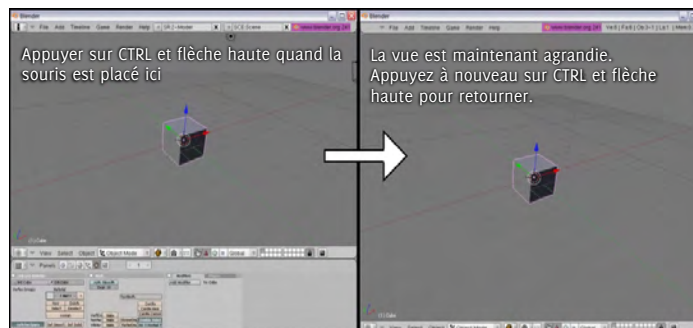
Déplacez votre souris sur la fenêtre 3D, et appuyez sur [Ctrl-Haut](flèche du haut)(ou Ctrl flèche du bas). Cela va mettre la fenêtre courante aux dimensions de la fenêtre Blender. Si vous appuyez à nouveau sur [Ctrl-Haut], la fenêtre va revenir à sa taille et sa position précédentes.

Continuons et faisons-le à présent :

- Maximisez la fenêtre 3D en utilisant [Ctrl-Haut]
- Appuyez sur [P] pour jouer la scène actuelle dans le GE (rien ne se produira)
- Appuyez sur [Echap] pour retourner au mode de modélisation
- Restorez la fenêtre 3D à sa taille originelle à nouveau en utilisant [Ctrl-Haut]

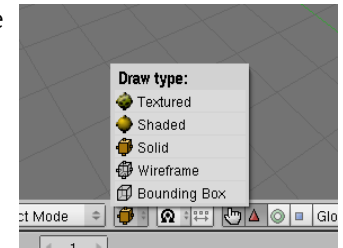
La séquence d'images ci-dessous montre les effets de l'utilisation des touches Maximiser/Restorer dans l'écran de configuration par défaut de Blender.

Maximisez les panneaux de Blender en utilisant [Ctrl-Haut].



Choisir le mode d'ombrage de la vue correct (ou Draw Type) pour le GE

Entrez à nouveau dans le mode GE en déplaçant votre souris sur la fenêtre 3D et appuyez sur [P]. Vous remarquerez que la scène dans le GE a l'air plate. Appuyez sur [Echap] pour retourner à Blender. Blender possède différents mode d'ombrages de la vue qui sont utiles pour différentes tâches. Le panneau pour changer l'ombrage est montré plus bas...



L'image en dessous montre à quoi ressemble la scène basique dans le GE, dans les différents modes d'ombrage de la vue

Le mode Solide ne prend pas en compte les lumières de la scène

Le mode Ombrée prend en compte les lumières de la scène

Le mode Texturé prend en compte les lumières de la scène, et montre également toutes textures appliquées dans la fenêtre d'affichage. Ce sera aussi proche de la vue dans le jeu que vous pouvez obtenir, et devrait toujours être sélectionné lorsque vous commencez un nouveau projet de GE.

Le meilleur affichage de la vue pour le GE est Texturé. Sélectionnez le mode d'ombrage Texturé dans la liste, et appuyez à nouveau sur [P].

Vous remarquerez que l'éclairage affecte maintenant l'environnement dans le GE, ce qui le rend plus réaliste. Souvenez-vous toujours de mettre cette option si vous jouez votre scène dans le GE et qu'elle vous semble plate.



Le panneau Game Logic principal

En dessous de la fenêtre 3D, vous verrez le panneau qui contient plusieurs boutons différents pour contrôler différents aspects de Blender.

Vous pouvez voir le panneau relatif au GE en cliquant sur l'icône du Pacman mauve, juste comme en dessous ou en pressant sur [F4].



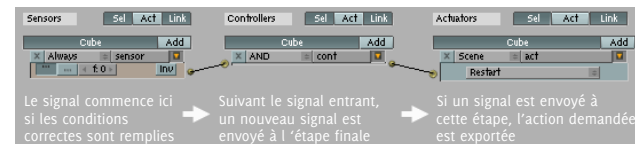
C'est dans ce panneau que vous contrôlerez tout ce qui se passera dans votre jeu.

Blender utilise un système visuel de cliquer-glisser pour créer des interactions basiques de jeu. Cela permet au GE d'être utilisé par des artistes 3D qui ne sauraient pas coder ou avoir l'aide d'un programmeur. Blender a également un langage de programmation, Python, qui peut être utilisé pour créer des interactions de jeu plus complexes.

Dans ce tutoriel, nous allons nous concentrer sur le système visuel de création de jeu. Lorsque vous avez compris les bases de l'utilisation du GE de Blender, vous pouvez suivre des tutoriels plus avancés montrant comment implanter des scripts Python pour créer des jeux plus complexes.

Contrôler visuellement le GE - Sensors, Actuators, Controller Logic Blocks

Le système du GE utilise les Logic Blocks comme un moyen visuel de mettre en place des interactions dans le jeu. Ces Logic Blocks peuvent être reliés ensemble visuellement pour permettre à des actions de jeu complexes de s'exécuter.



Il y a trois différents types de Logic Blocks - Sensors, Controllers et Actuators - chacun avec un nombre de sous-types différents.

Sensors (Capteurs)

Un Sensor détectera une forme d'entrée. Cette entrée peut être n'importe quoi à partir d'une touche, d'une manette de jeu, d'une minuterie qui déclenche tout seul une mise à jour de l'écran (ou de frame) du jeu.

Controllers (Contrôleurs)

Les Controllers sont utilisés pour relier les Sensors aux Actuators. Ils permettent quelques contrôles complexes supplémentaires sur la façon dont les sensors et les actuators interagissent entre eux.

Actuators

Un actuator va lancer vraiment une action dans le jeu. Cela peut inclure bouger un objet dans une scène, jouer une animation, ou jouer un effet sonore.

Configuration d'une séquence basique de Logic Block Sensor, Controller, Actuator.

Nous allons maintenant mettre en place un système très basique dans le panneau de jeu en ajoutant et en connectant un sensor, un controller et un actuator ensemble.

Panneau du GE : Soyez sûr que le panneau de Jeu Logique est visible (cliquez sur le Pacman mauve dans la fenêtre des boutons ou appuyez sur [F4]), et re-sélectionnez le cube dans la scène 3D.

En-dessous de chacune des 3 principales sections, vous verrez le nom de l'objet sélectionné, et un bouton Add. Cliquez sur ce bouton Add pour chacune des 3 sections: Sensor, Controller and Actuator.

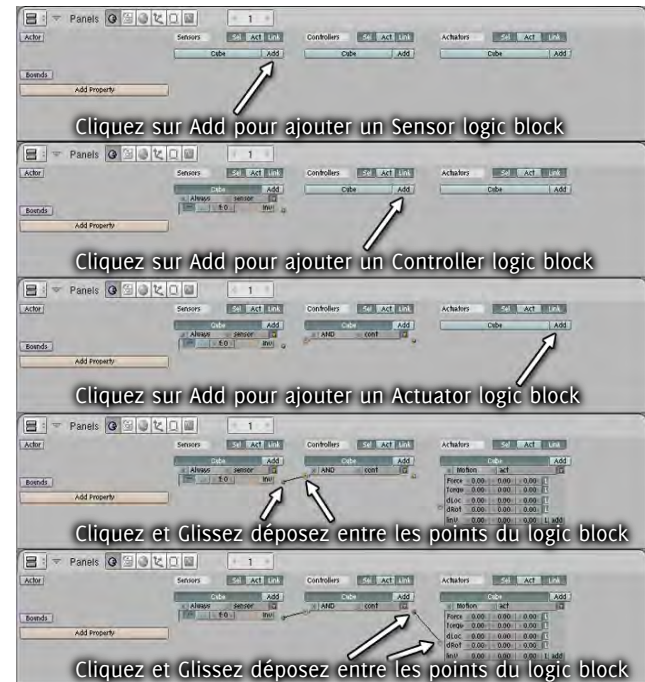
Nous allons maintenant connecter ce système ensemble. Cliquez et tirez à partir du petit cercle au bout du premier capteur jusqu'au cercle au début du controller. Ensuite cliquer et tirez du cercle au bout du controller au cercle au début de l'actuator.

La séquence d'images sur la droite montre les étapes impliquées dans la mise en place d'une simple chaîne Sensor, Controller, Actuator et leurs connections ensemble.

Appuyez sur [P] pour lancer le jeu maintenant. Vous remarquerez que, bien que nous ayons ajouté quelques contrôles au GE, rien ne semble encore se produire. Cela sera expliqué dans la prochaine section. Appuyez sur [Esc] pour retourner dans Blender.

Décomposition des événements dans le système GE

Examinerons maintenant ce qui se passe avec notre système GE nouvellement créé. Le sensor est un Always timer. Cela fera sortir un signal à chaque frame en gardant le contrôleur lié continuellement activé pendant la durée du jeu.



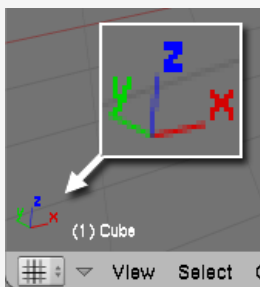
Le contrôleur est un AND. Quand il a juste un sensor input actif (comme dans le cas actuel), il appellera automatiquement l'Actuator connecté.

L'Actuator contrôle les aspects de Mouvement de l'objet choisi. À cause du sensor Always connecté au contrôleur, cet Actuator sera appelé chaque frame.

Si nous appuyons [P] maintenant, l'Actuator de Mouvement est appelé à chaque frame, mais parce que toutes ses valeurs sont mises à zéro, il ne déplacera pas l'objet dans le GE. Appuyez [Esc] de nouveau pour retourner à Blender.

Garder les axes en tête

Pour savoir dans quelles directions sont X, Y et Z, gardez un oeil sur le repère en bas à gauche de la vue 3d.



Faire bouger le cube par défaut, sans utiliser le moteur physique

Nous allons d'abord utiliser une manipulation directe pour déplacer le cube dans la scène. Plus tard, nous configurerons une scène similaire, mais en utilisant le moteur physique pour déplacer le cube au sein de l'environnement. En utilisant le moteur physique intégré (appelé Bullet) des interactions de scène plus complexes seront réalisées automatiquement comme les collisions ou la gravité.

Jetez un oeil sur l'actuator Motion, surtout aux 3 boîtes numériques à côté du label dLoc. Chacune des 3 boîtes de cette zone dLoc peut être utilisée pour spécifier un changement de position de l'objet le long des axes X, Y ou Z respectivement.



Mettez la valeur numérique dLoc (axe Y soit "avancer") à 0.10.

Appuyez maintenant sur [P]. Vous remarquerez que le cube bouge continuellement le long de l'axe Y. Appuyez sur [Esc] pour quitter le Game Engine. Appuyez de nouveau sur [P] et vous verrez qu'exactement la même séquence d'événements se produit dans le GE. Appuyez sur [Esc] pour retourner de nouveau dans Blender.

Pour récapituler ce qui se passe dans le GE - le timer (Sensor) envoie un signal à chaque frame vers le Controller du cube. Celui-ci envoie alors le signal à l'Actuator du cube, déplace le cube de 0.1 unités sur l'axe Y. Comme il est appelé à chaque frame, on a l'impression que le cube se déplace en continu. Si vous laissez tourner le jeu, le cube va glisser en fin de compte à l'infini. C'est au moins aussi passionnant que de regarder l'herbe pousser, alors, continuons!

Contrôler le cube avec les touches fléchées

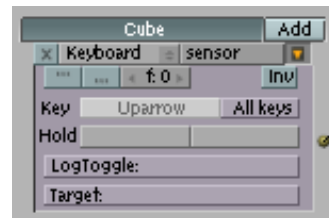
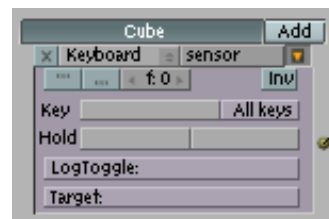
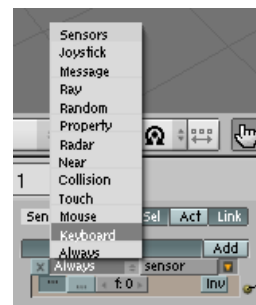
Dans l'exemple à droite, le timer est responsable du déplacement du cube. Nous allons changer cela pour que ce soit une touche qui le contrôle.

Dans le Sensor Always, cliquez sur le > à côté du label Always. Cela va nous donner la liste des types de Sensor disponibles (voir fig ci-dessous).

Sélectionnez Keyboard dans la liste. Le Sensor change alors pour montrer les options Keyboard (de touches).

Ce nouveau panneau Sensor va nous permettre de choisir une touche qui devra être appuyée avant qu'un signal ne soit envoyé au Contrôleur, qui le passera à son tour à l'Actuator. Cliquez sur le bouton à côté du label Key, et quand il vous dit d'appuyer sur une touche, appuyez sur la touche [flèche haute].

Appuyez sur [P] pour lancer à nouveau le jeu. Le cube ne se déplace plus tout seul. Appuyez sur la touche [flèche haute], et il va commencer à avancer. Quand vous arrêter de presser la touche, le cube s'arrête. Appuyez sur [flèche haute] à nouveau pour déplacer le cube. Appuyez sur [Echap] pour retourner dans Blender.



Ajouter des contrôles clavier supplémentaires

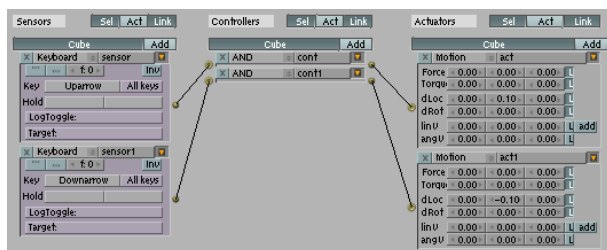
Nous allons ajouter la possibilité de faire reculer le cube, ainsi que de le faire pivoter, pour vous permettre de le déplacer partout dans l'environnement 3D.

Pour ajouter la possibilité de reculer, nous allons ajouter un nouveau (mais pratiquement identique) jeu de briques logiques. Cliquez sur le bouton Add des zones Sensor, Controller and Actuator pour créer 3 nouvelles briques dans le panneau GE. Connectez-les comme précédemment.

NOTE - Si vous le voulez, vous pouvez utiliser [Ctrl+flèche haute] pour maximiser et restaurer le panneau du game logic quand vous travaillez dedans, étant donné qu'il va vite sembler désordonné.

Passez le nouveau Sensor en type keyboard, et configurez-le pour utiliser la touche [flèche basse]. Dans l'actuator Motion, changez la valeur Y (2ème colonne) de la zone dLoc à -0.1.

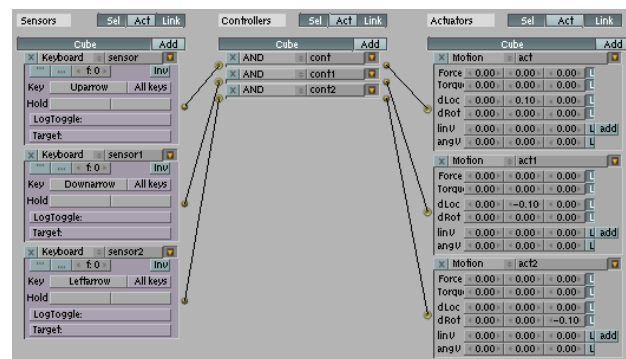
Appuyez sur [P] pour lancer le jeu. Si les logic blocks sont configurés correctement, vous devriez être capable maintenant de déplacer le cube d'avant en arrière simplement avec les touches [flèche haute] et [flèche basse].



Pour compléter cette partie "sans physics" du tutoriel, nous allons ajouter la possibilité de pivoter l'objet, ainsi vous serez capable de conduire votre modèle dans l'environnement 3D.

Ajoutez encore une fois 3 nouvelles briques logiques, connectez-les, et changez le Sensor type en keyboard pour utiliser la touche [flèche gauche].

Maintenant, nous allons configurer le Motion Actuator pour pivoter le cube, c'est-à-dire le faire pivoter autour de l'axe Z. Dans la zone dRot, changez la valeur Z (3ème colonne) à -0.1.



Appuyez sur [P] pour lancer le jeu. Quand vous cliquez la touche [flèche gauche], le cube va pivoter. Quand vous appuyez sur la flèche [Haut], le cube va avancer dans cette direction. Appuyez sur [Esc] pour retourner dans Blender et nous allons ajouter la possibilité de tourner dans l'autre sens.

Comme avant, ajoutez 3 nouvelles briques, connectez-les et changez le type du Sensor en keyboard.

Configurez l'entrée du clavier pour utiliser la flèche [Droite] et la valeur Z (3ème colonne) de la section dRot du Motion Actuator à 0.1.

Appuyez encore sur [P] pour lancer le jeu. Vous pouvez à présent diriger le cube dans l'environnement 3D en utilisant les touches fléchées. Remarquez que vous pouvez ajouter ce game logic à n'importe quel modèle dans Blender (peu importe la forme ou taille) et il se déplacera exactement comme le cube. Appuyez sur [Esc] pour retourner dans Blender.

Un peu de rangement

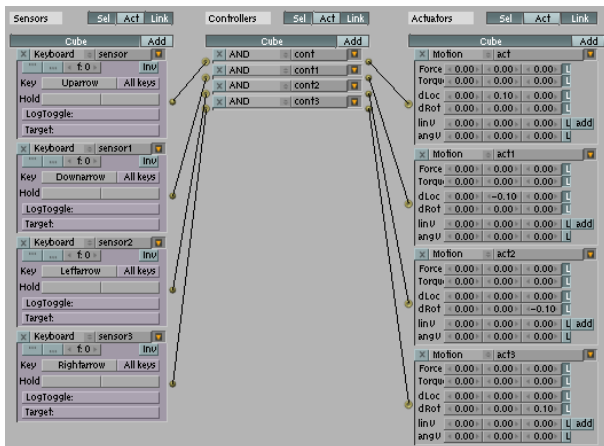
Nous allons maintenant nommer certains des Sensors et les minimiser, pour garder le panneau Game aussi léger et en ordre que possible. Cela n'affectera pas l'action des logics, mais cela facilitera la lecture et la manipulation.

Appuyez sur [Ctrl+Flèche haute] dans le panneau jeu pour le mettre en plein écran.

Dans le premier panneau Sensor, à gauche du type "Keyboard", vous verrez une zone où le texte "Sensor" apparaît. Cela peut être changé pour un autre mot plus explicite. Changez le texte en "up key". Ensuite, appuyez sur le bouton flèche à côté pour minimiser le panneau Sensor.

Répétez cela pour les autres touches. Faites-le également pour les Actuators, en leur donnant un titre explicite (par exemple "avancer", "tourner gauche", etc...)

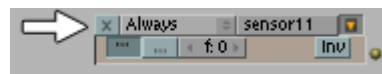
Appuyez sur [Ctrl+Up] à nouveau pour restaurer la taille originale du panneau game.



Comme vous pouvez le voir, avoir les différents blocs nommés ainsi nous facilitera la vie pour la suite.

Supprimer des logic blocks et des connections

Pour supprimer des logic blocks, appuyez sur le bouton X dans le coin au-dessus à gauche du logic block. Pour supprimer



mer une connection entre 2 logic blocks, placez la souris sur la ligne de connection et appuyez sur la touche [Del].

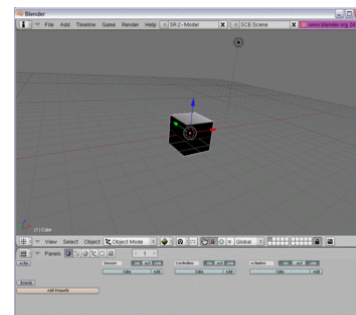
Faire avancer le cube par défaut en utilisant le moteur physique

Une des fonctions les plus puissantes du GE de Blender est le moteur de physique incorporé (Bullet).

En utilisant des forces pour déplacer un objet autour de la scène, le système des physics va manipuler automatiquement des interactions complexes, comme détecter les collisions avec d'autres objets dans la scène. Pour beaucoup de types de jeu, utiliser les Physics résoudra beaucoup de cas plus complexes, mais cela demande beaucoup plus de travail de mise en place.

Nous allons créer maintenant à partir de zéro un jeu basique utilisant les physics. La scène consistera en une sphère que nous allons déplacer autour de la scène en utilisant des forces physiques.

Avant que nous ne commençons, réinitialisez la scène dans Blender pour revenir à celle par défaut, et mettez la vue 3D comme sur l'image à droite.

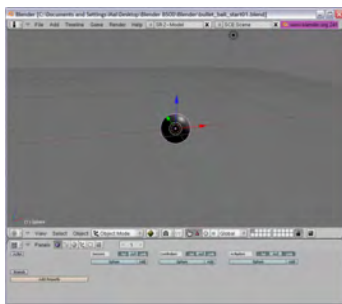


Configurez Blender pour qu'il ressemble à ça pour la suite

- Sélectionnez le menu File » New puis Erase all
- Pivotez la vue en utilisant le bouton du milieu de la souris (BMS)
- Passez en mode perspective, menu View » Perspective (ou pavé num [5])
- Passez en mode texturé ombré (Textured) ou appuyez sur [Alt+Z]
- Ouvrez le panneau Game (/Logic)

Configuration de la scène: pour créer la scène de départ du jeu, supprimez le cube en appuyant sur [Del].

- Ajoutez une sphère en faisant Add » Mesh » UVSphere (La barre d'espace affiche aussi ce menu).
- Lorsque le panneau s'affiche, laissez les valeurs par défaut à 32 segments et 32 rings
- Sortez du mode d'Editon en appuyant sur [Tab], pour retourner dans le mode Objet.
- Appuyez aussi sur [Alt+R] pour réinitialiser la rotation de l'objet (l'objet doit être sélectionné).



Créer un modèle physique dans le GE

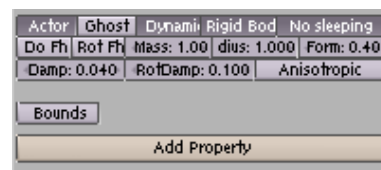
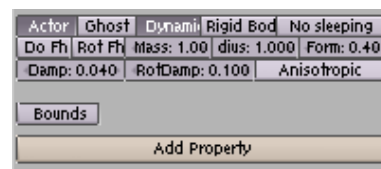
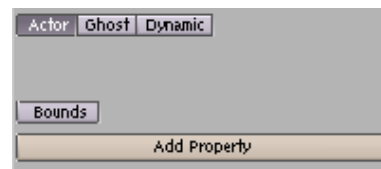
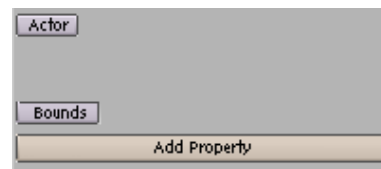
Dans le panneau de jeu, vous verrez un bouton Actor désactivé.

Cliquez dessus, et d'autres éléments apparaîtront suivant le modèle sélectionné.

Sélectionnez l'option dynamique - Cela indiquera au GE qu'il s'agit un objet physique. Plusieurs options vont apparaître maintenant.

Sélectionnez les options suivantes...

- Rigid Body: L'objet physique tournera correctement en utilisant le physics engine du GE. Si ce bouton n'est pas sélectionné, l'objet pourra bouger, mais pas tourner.
- No sleeping: L'objet physique ne sera jamais désactivé (également connu comme "sleeping").



Maintenant, appuyez sur [P] pour entrer dans le GE. Vous observerez que, bien que nous n'ayons pas ajouté de logic blocks, la balle commence à bouger. La gravité affecte notre balle, donc elle tombe. Ceci montre une des possibilités de l'interaction dans un monde physique. Appuyez sur [Echap] pour retourner à Blender. Appuyez encore sur [P] et vous observerez encore la même chose. Appuyez sur [Echap] pour revenir à Blender.

Nous avons besoin d'ajouter quelque chose pour que la balle tombe dessus, par exemple un objet sol.

- Ajoutez un plan à la scène en faisant Add » Mesh » Plane.
- Appuyez sur [Tab] pour sortir du mode Edit et revenir au mode Objet.
- Appuyez sur [Alt+R] pour réinitialiser la rotation du modèle.

Utiliser le manipulateur Transformation 3D pour déplacer le plan sous la sphère et appuyez sur [P]. Vous remarquerez que la sphère chute maintenant à cause de la gravité mais atterrit sur le plan en dessous et y reste.

Nous allons agrandir le plan [S], ainsi nous aurons de la place pour y déplacer la balle.

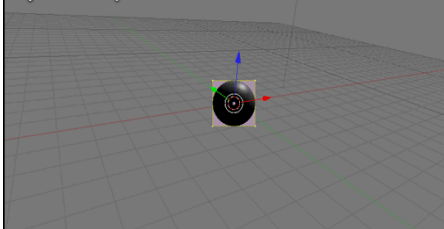
Changez le manipulateur Transformation 3D du mode déplacement au mode redimensionnement.

Déplacez les poignées de redimensionnement et agrandissez le plan de façon à ce qu'il soit 10 fois plus grand.

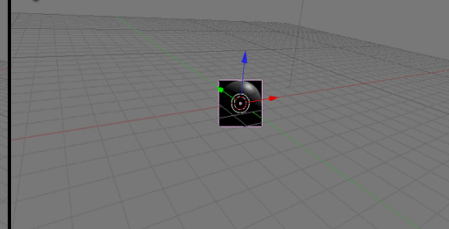
La séquence d'images à droite illustre les différentes étapes.

Ajouter et placer un objet sol (voir les images à droite)

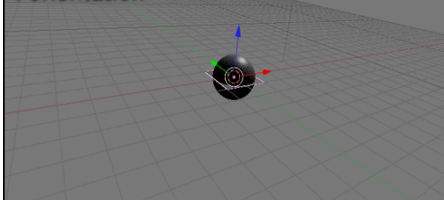
Ajout du plan add->Mesh->Plane



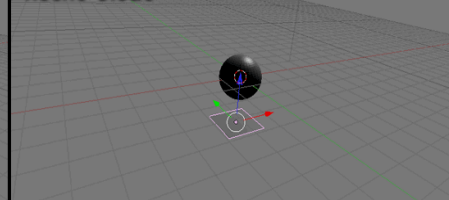
Quittez le mode d'édition avec Tab



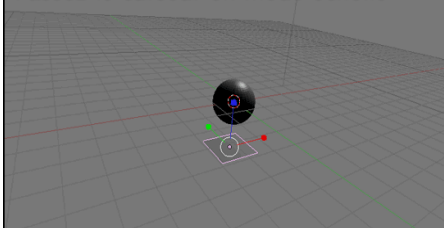
Appuyez sur Alt+R pour réinitialiser l'orientation



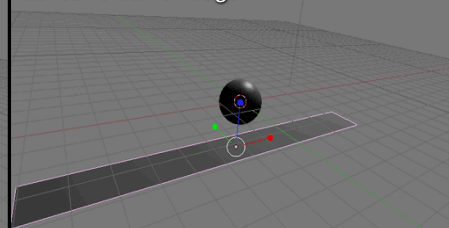
Déplacez le plan vers le bas avec la flèche bleue



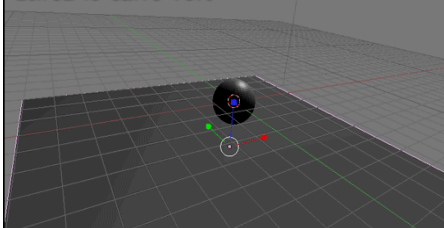
Passez le curseur en mode échelle



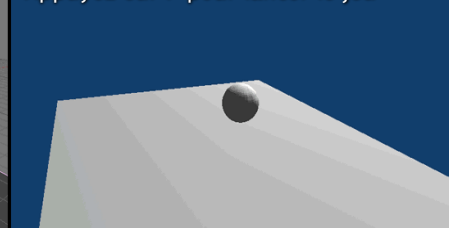
Étirez le carré rouge



Étirez le carré vert



Appuyez sur P pour lancer le jeu



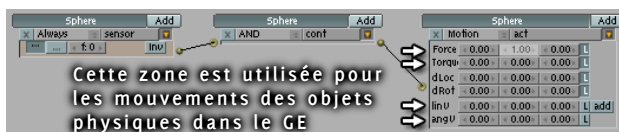
Déplacer l'objet physique dans le GE

Nous allons maintenant appliquer des forces physiques à la sphère pour la faire bouger dans l'environnement 3D.

IMPORTANT - Assurez-vous que la sphère est sélectionnée. Si vous venez tout juste de créer le plan du sol, il sera encore sélectionné, donc vous devez cliquer sur la sphère pour la sélectionner.

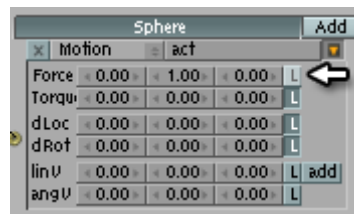
Ajoutez un nouveau Sensor, Controller et Actuator dans le panneau Game, et connectez-les ensemble en cliquant et tirant entre les points.

Dans l'actuator Motion, nous aurons besoin de mettre les valeurs dans la section Force de manière à déplacer l'objet dans la scène. Dans la section Force, changez la valeur Y (2ème colonne) à 1.



Appuyez maintenant sur [P]. Vous verrez à présent la ball tomber sur le sol/plan et commencer à rouler.

Après avoir rouler sur une certaine distance, vous remarquerez que la balle commencer à rouler en arrière sur elle-même. Cela est dû au fait que nous avons appliqué la force localement, le long de l'axe Y de la balle. Quand la balle roule, l'axe Y roule aussi, comme vu ci-dessous. Appuyez sur [Esc] pour retourner dans Blender.



verrez que la balle continue à se déplacer dans la bonne direction.

Vous n'avez peut-être pas compris comment le moteur physique fonctionne.

Quand nous appliquons une force latérale à la balle, elle va commencer à rouler. Ce roulement est causé par la friction entre la surface de la balle et le sol.

Ainsi, quand la balle atteint l'extrémité du plan, elle va basculer de façon réaliste au bord de l'objet et tomber.

Voilà quelques-uns des avantages à utiliser les physics dans le GE.

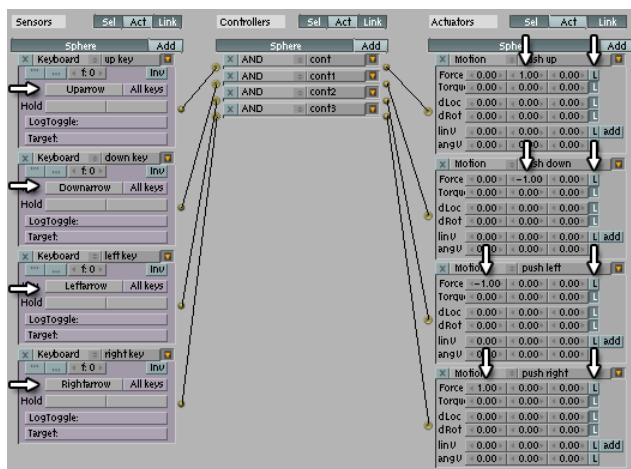
Appuyez sur [Esc] pour retourner dans Blender.

Contrôler la sphère en utilisant les touches fléchées

Nous allons maintenant prendre le contrôle de la sphère avec les flèches.

Changez le Sensor Always en Keyboard, et configurez-le pour utiliser la touche [Up] (haut). Maintenant, ajoutez et connectez des sensors, controllers et actuators, ensemble

- Le Sensor keyboard [Down] contrôle le motion actuator avec un -1 sur Y (2ème) de la section Force (avec L de Local désélectionné)
- Le Sensor keyboard [Left] contrôle le motion actuator avec un -1 sur X (1er) de la section Force (avec L de Local désélectionné)
- # Le Sensor keyboard [Right] contrôle le motion actuator avec un -1 sur X (1er) de la section Force (avec L de Local désélectionné)



Si vous appuyez encore sur [P], vous serez capable de contrôler la balle et de la déplacer autour du plan. Appuyez sur [Echap] pour retourner dans Blender.

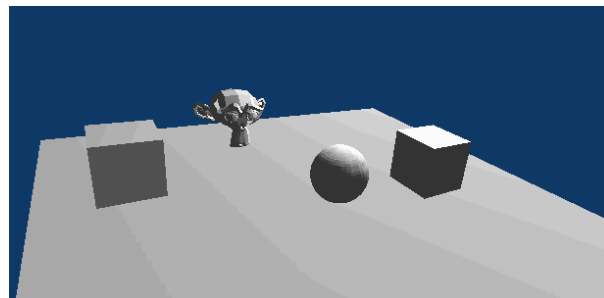
Pour rendre le panneau Game plus lisible, donnez aux différents sensors et actuators des noms explicites, comme "haut" pour le premier sensor et "pousser haut" au premier actuator.

Essayez également de changer les valeurs dans tous les motion actuator, de 1 à 2 et de -1 à -2. Appuyez sur [P] à présent, et vous verrez la différence dans la vitesse de la balle quand vous utilisez les touches fléchées.

Actuellement le système de physique supposera que l'ajout d'un nouveau Rigid Body aura une forme de collision sphérique. Appuyez [Esc] pour retourner à Blender.

Ajouter quelques obstacles dans le niveau

Ajoutez un cube dans l'environnement, Add >> Mesh >> Cube. Appuyez sur [Tab] pour quitter le mode Edition et revenir au mode Objet. Appuyez sur [Alt+R] pour réinitialiser la rotation de l'objet et tirer les flèches du manipulateur Transformation 3D pour placer la boîte quelque part à la surface du plan. Répétez cette étape pour ajouter quelques autres objets en plus sur le plan, en incluant des cylindres et des Suzannes.



Maintenant, appuyez sur [P]. Vous verrez que la balle va automatiquement taper et rebondir sur les objets que vous venez d'ajouter. Encore une fois, ceci est un des avantages de l'utilisation des physics dans le GE. Appuyez sur [Echap] pour retourner dans Blender. Vous pouvez aussi ajouter des plans et les redimensionner pour faire des rampes et des tremplins.

Si vous avez de l'expérience en modélisation dans Blender, vous pouvez passer un peu de temps à créer un niveau de jeu plus complexe.

Si vous n'avez pas d'expérience en modélisation, espérons que ce tutoriel vous donnera de l'intérêt pour Blender et l'envie d'en apprendre davantage, y compris dans la manière de modéliser des objets. Vous pouvez voir quelques liens sur l'édition à la fin de ce tutoriel.

Créer quelques objets physiques

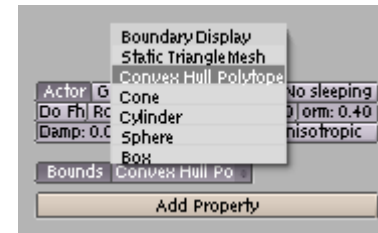
Sélectionnez le cube que vous avez ajouté dans la nouvelle scène. Dans le panneau Game, sélectionnez les options suivantes, les mêmes que celles qui ont été affectées à la sphère principale (à l'exception de No Sleeping qui n'est pas sélectionné cette fois, pour permettre aux objets de ralentir et de se reposer / dormir - quand un objet est "endormi" (sleeping), il prend moins de temps de calcul dans le système de physics).

- Actor
- Dynamic
- Rigid Body

Appuyez à présent sur [P] pour lancer le jeu et déplacer la sphère principale sur le cube. Vous allez voir que le cube va être éjecté sur le côté.

Cependant, la boîte se déplace d'une manière très bizarre - il se comporte en fait comme s'il était une sphère.

Actuellement, le système des physics va estimer qu'un rigid body nouvellement ajouté a une forme sphérique pour les collisions. Appuyez sur [Echap] pour retourner dans Blender.



Vous remarquerez qu'il y a un bouton Bounds sous la zone de l'Actor. Cliquez sur ce bouton et ajouter un nouveau menu va apparaître avec Box comme paramètres par défaut.

Dans le cas du cube, un type de collision Box convient parfaitement. Cependant si vous avez une forme plus complexe, vous sélectionnez sans doute l'option Convex Hull Polytope.

Sélectionnez quelques uns des autres objets que vous avez ajouté à la scène et effectuez les mêmes étapes que précédemment, sélectionnez Convex Hull Polytope comme type de limite pour l'objet.

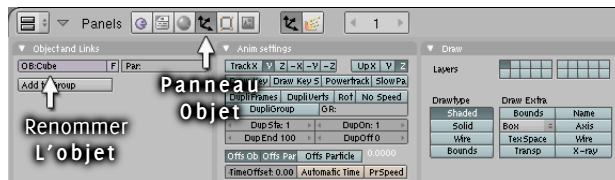
Appuyez sur [P] encore pour jouer le niveau en cours, et rouler sur les différents objets physiques pour les bousculer hors du chemin. Appuyez sur [Echap] pour retourner dans Blender.

Renommer les objets

Quand vous ajoutez un nouvel objet, Blender lui assigne un nom par défaut (par ex. Cube ou Cube.001 si Cube existe déjà).

C'est une bonne habitude de renommer vos objets en utilisant des termes plus explicites, comme "joueur", caisse, camionnette... Cela rendra votre scène plus facile à comprendre quand elle deviendra plus complexe, ainsi que de la rendre compréhensible pour les autres personnes qui la verront.

Pour renommer un objet, vous pouvez sélectionner le panneau Objet et changer le nom dans la zone OB:



Vous pouvez aussi le renommer dans le panneau Editing.



Complément au tutoriel sur les bases du GE

Félicitations pour avoir suivi ce tutoriel sur le Blender Game Engine ! Vous devriez avoir maintenant un aperçu général des bases pour utiliser le GE. Vous avez l'expérience pratique de...

- Connecter les sensors, contrôleurs et actuators dans le panneau jeu
- Utiliser l'actuator motion pour déplacer les objets directement
- Utiliser l'actuator motion pour déplacer les objets en utilisant les forces physiques
- Prendre le contrôle des objets avec le clavier
- Créer une scène simple d'un jeu 3D
- Créer des nouveaux objets physiques dans le GE

Avec les connaissances que vous avez apprises jusqu'ici, vous êtes capable d'étendre ce simple environnement comme si vous appreniez davantage sur la modélisation dans Blender.

A ce stade, vous voudrez peut-être recréer la scène finale encore une fois, en partant de zéro, pour voir jusqu'où vous êtes capable d'aller sans relire les instructions. Si vous êtes capable de recréer tout de mémoire, vous êtes en bonne voie pour devenir un vrai utilisateur de la puissance du Blender GE !

Les chapitres supplémentaires ci-dessous couvriront certains problèmes plus complexes, tels que le faire sauter la balle, et l'ajout de matériaux à la scène.

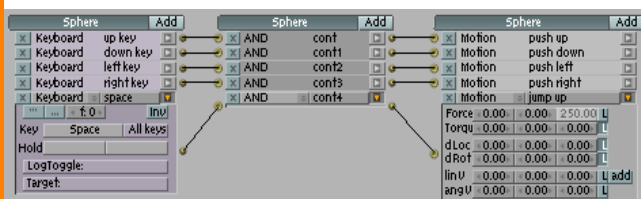
Créer des niveaux de jeu plus complexes et des interactions :

Faire sauter la balle

Nous allons faire sauter la balle lorsque l'on presse la touche [Espace].

Ajouter des nouveaux sensor, controller et actuator à la scène et relier-les. Changez le sensor Always en Keyboard et configurez-le pour utiliser la touche [Espace].

Dans la section Force du Motion actuator, mettez la 3ème valeur (l'axe Z/haut) à 250. Cliquez aussi sur le L pour l'enlever, de manière à ce que la force soit appliquée le long de la direc-



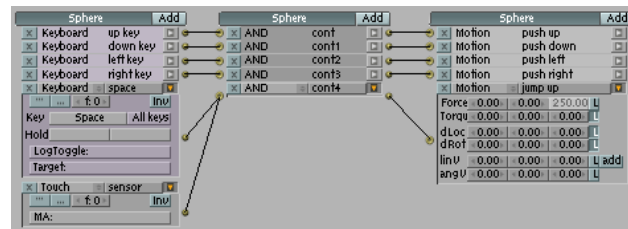
tion global et non dans la direction local de Z.

Cela donnera à la balle un petit coup, qui fera comme si elle sautait en l'air.

Appuyez sur [P] et dans le jeu, taper sur [Espace]. Vous remarquerez que la balle sautera directement en l'air vers le haut. D'ailleurs, si vous continuez à maintenir la touche, la balle va continuer à monter en l'air.

Nous avons besoin d'ajouter une règle supplémentaire - la balle peut sauter seulement si elle touche le sol. Pour cela, nous utiliserons un sensor Touch. Ajoutez juste un sensor supplémentaire à la scène et changez le en type Touch. Main-

tenant, connectez ce sensor au contrôleur auquel est déjà connecté le sensor de la touche [Espace].



Le contrôleur va seulement envoyé un signal au Motion actuator connecté (pour l'action saut) quand EN MÊME TEMPS la touche [Espace] est pressé ET la balle cogne contre un autre objet, comme le sol.

Redémarrer le jeu quand un but est atteint

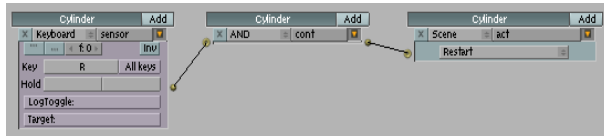
Quand la balle touche un objet particulier, nous allons faire redémarrer le niveau.

- Ajoutez un nouvel objet à la scène (dans ce cas, nous allons mettre un cylindre) et placez le quelque part près de la sphère principale (mais ne la touchant pas).
- Assurez-vous que le nouvel objet soit sélectionné (et non la sphère principale).

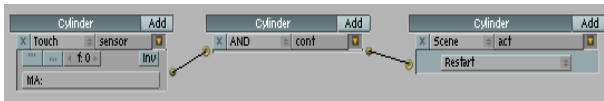


- Ouvrez le panneau Logic panel, ajoutez sensor, controller et actuator et reliez-les entre eux.
- Changez le type d'Actuator en Scene. Les paramètres par défaut de l'actuator Scene sont réinitialisés.

- Maintenant, appuyez sur [P] pour démarrer le jeu. Il ne se passera rien en apparence. C'est parce que le sensor Always est continuellement en train de déclencher le sensor Restart Scene.
- Appuyez sur [Echap] pour quitter le jeu.
- Changez le type de sensor en Keyboard, et configurez le pour utiliser la touche [R].
- Appuyez sur [P] pour démarrer le jeu à nouveau, et vous pourrez déplacer la sphère.



- Appuyer sur [R] redémarrera le jeu et vous permettra de continuer.
- Appuyez sur [Echap] pour quitter le jeu.
- Enfin, changez encore le type du sensor en Touch.



- Appuyez sur [P] pour démarrer à nouveau le jeu. La scène va redémarrer maintenant quand vous roulez dans le cylindre avec la sphère.
- Appuyez sur [Echap] pour quitter le jeu.

Ceci illustre quelques bases de la gestion de scènes du GE. Votre jeu peut aussi mener à une autre scène (non expliqué dans ce tutoriel) qui peut comporter une séquence "jeu ga-

gné" ou "jeu perdu"; ou bien la scène peut être un niveau supplémentaire du jeu.

Collecter des objets dans le niveau

- Nous allons maintenant ajouter un objet que le joueur peut collecter quand il se déplace tout près de lui.
- Ajoutez une sphère à la scène et placez-la à une distance raisonnable de la sphère principale.
- Ajoutez et connectez des Sensor, Controller et Actuator.
- Changez le type Sensor en Near.
- Changez le type de l'actuator en Edit Object.



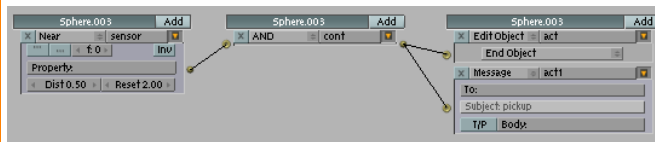
- Pour Edit Object, changez-le d'Add Object en End Object.
- Activez aussi les boutons Actor et Ghost. L'objet Ghost signifie que les autres objets (y compris le joueur principal) NE sera PAS capable d'entrer en collision avec l'objet.
- Appuyez sur [P] pour démarrer le jeu. Maintenant quand vous vous déplacez près de l'objet, il disparaîtra (en mettant fin lui-même).
- Appuyez sur [Echap] pour revenir dans blender.

Compter les objets récoltés

Quand un objet est collecté, nous mettrons à jour la valeur d'une propriété pour refléter le nombre total de choses collectées jusqu'à présent.

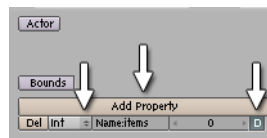
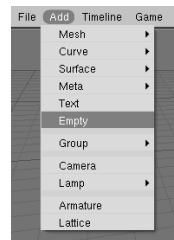
Nous allons utiliser l'actuator Message pour envoyer un signal à un autre objet dans la scène qui aura un sensor Message. Ce sensor va déclencher un actuator qui incrémentera la valeur d'une propriété.

- Ajoutez un autre actuator à l'objet collecté et changez son type en Message.
- Connectez cet actuator Message au controller existant.
- Changez le nom en quelque chose comme "pickup". Il est important de se rappeler ce nom, qui sera nécessaire pour le sensor Message plus tard.



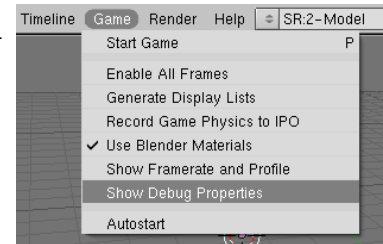
Quand le sensor Near est activé, chacun de ces actuators va être déclenché et le message sera envoyé à tous les objets de la scène.

- Nous allons utiliser un autre objet pour stocker les informations du compteur des objets collectés. Pour cette tâche, un Empty sera utile - c'est un objet qui existera dans la scène, mais qui n'a aucune géométrie et qui ne sera pas visible dans le GE.
- Ajoutez un Empty à la scène.
- Quand cet objet est sélectionné, cliquez sur le bouton

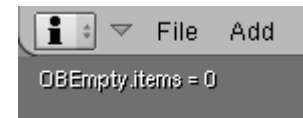


ton Add Property dans le panneau du Game (voir image Add Property).

- Changez le nom du l'objet nouvellement ajouté pour "items" et changez le type Float en Int (Integer ou nombre entier, par exemple 0, 1, 2, 3). C'est ici que nous allons stocker le nombre d'items collectés.



- Enfin, appuyez le bouton D (pour "Debug") à la droite de la propriété. Cela vous permettra de voir la valeur de la propriété en jeu. Pour la voir, sélectionnez l'option Show Debug Properties du menu.



- Appuyez [P] pour lancer le jeu, et vous remarquerez un texte dans le coin supérieur gauche de l'écran 3d, montrant la valeur de la propriété des items (actuellement à 0).
- Appuyez sur [Echap] pour retourner dans Blender.
- Ajoutez une séquence Sensor, Controller et Actuator Logic Block sur l'empty et connectez-les ensemble.
- Changez le type du sensor en Message et mettez le nom à "pickup" (le même nom que vous avez mis pour le sensor Message de l'objet collecté).
- Changez le type de l'actuator en Property et changez Assign pour Add.
- Changez le nom de Prop en "items" (le nom de la propriété ajoutée) et mettez la valeur à 1.



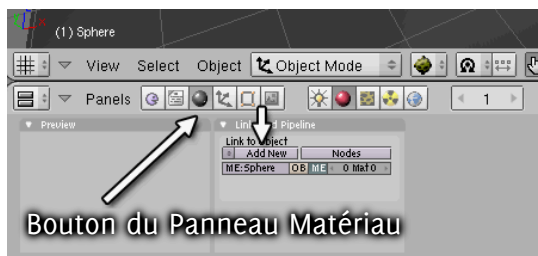
Maintenant, appuyez sur [P] pour lancer le jeu et attraper un objet - vous remarquerez que, quand vous le récoltez, la valeur de la propriété augmente. Appuyez sur [Echap] pour retourner dans Blender.

Vous pouvez utiliser le résultat de cette propriété pour affecter votre jeu, comme le redémarrer ou aller à une nouvelle scène quand un certain nombre d'objets est récolté.

Ajouter de la couleur aux niveaux en utilisant les matériaux

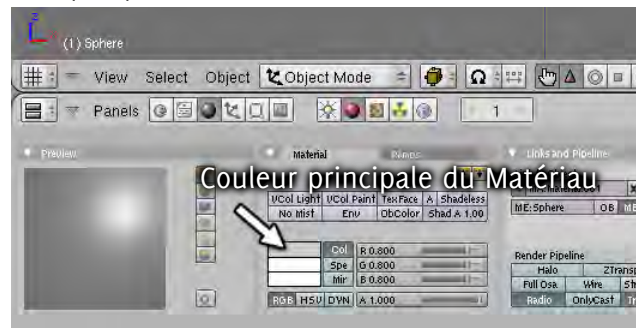
Jusqu'à présent, les objets ajoutés avaient la couleur grise par défaut. Dans le but de changer l'apparence de base de la scène, nous allons changer à présent la couleur des objets de la scène en créant des nouveaux matériaux pour eux.

- Ouvrez le panneau matériel en cliquant sur la sphère grise dans le panneau ou en appuyant sur [F5] comme montré plus bas.
- Sélectionnez la sphère principale dans la vue 3D avec un clic droit.

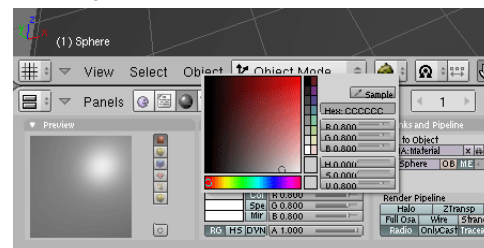


- Cliquez que le bouton Add New. Cela va ajouter un nouveau matériau à la sphère. Un ensemble de panneau plus complexes va apparaître. Pour le moment, nous allons seulement changer la couleur du matériau.

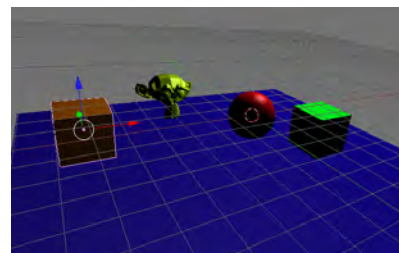
- Cliquez dans la zone sous Col, qui indique la couleur principale du matériau.



- Un sélecteur de couleur apparaît alors. Utilisez-le pour choisir une couleur rouge et ensuite déplacez le curseur de la souris hors du sélecteur. La sphère apparaît maintenant rouge dans la vue 3D.



Répétez ce processus pour les autres objets de la scène pour qu'ils soient tous de couleurs différentes.



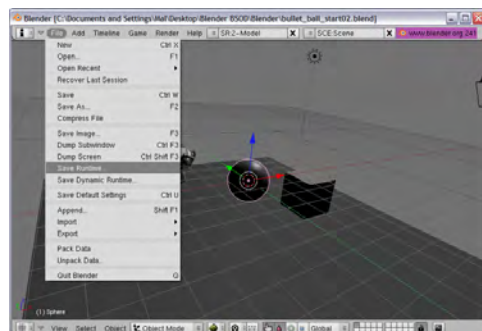
NOTE

Vous pourriez devoir inclure quelques autres fichiers en plus avec l'exécutable. Copiez le fichier dans un nouveau dossier et lancez-le. Si cela vous donne une erreur disant qu'il manque un fichier, copiez ce fichier (certainement un fichier .dll) dans le même dossier. Ces .dll sont en général localisés dans le dossier d'installation par défaut de Blender. Continuez ce processus jusqu'à ce que le jeu tourne. Vous pourrez alors distribuer ces fichiers à vos collègues. Assurez-vous seulement que les .dll et le .exe soient dans le même dossier.

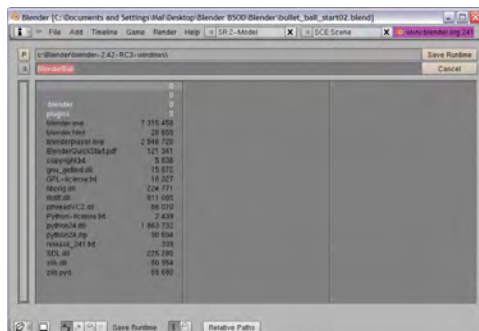
Faire une version indépendante du jeu

Blender vous permet de créer une version indépendante de votre jeu pour le distribuer aux collègues, sans qu'ils ne doivent installer Blender. Votre jeu se lancera automatiquement quand le programme se lance

Dans Blender, sélectionnez File » Save Runtime.



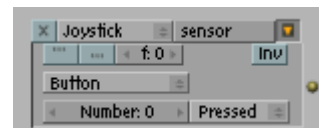
Dans l'écran de sauvegarde, entrez un nom pour l'exécutable du jeu (par exemple ball_game). Cela créera un exécutable ball_game dans le dossier que vous pourrez distribuer à vos amis.



Aperçu de tous les Sensor, Controller et Actuator Logic Blocks

Sensors

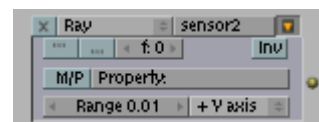
Joystick - se déclenche quand un bouton de joystick est pressé ou quand un joystick est actionné le long d'une certaine direction (gauche/droite, haut/bas, etc.)



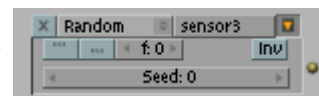
Message - se déclenche quand un message est reçu. Vous pouvez envoyer des messages à d'autres objets en utilisant un actuator Message.



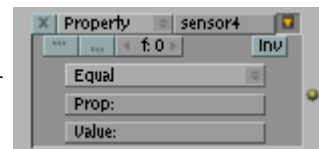
Ray - Ceci se déclenchera quand un objet est détecté le long d'un certain axe. Vous pouvez contrôler en plus si l'objet détecté un certain matériau ou une valeur propre.



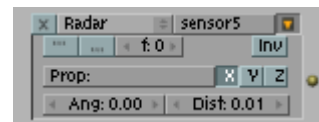
Random - se déclenche aléatoirement - change l'algorithme pour une séquence différente de nombres (ou utilisez python pour un vrai générateur aléatoire).



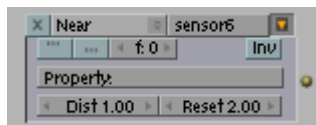
Property - se déclenche quand une propriété change, se trouve entre certaines valeurs minima et maxima, ou est égal ou différent d'une certaine valeur.



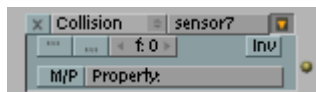
Radar - se déclenche quand un objet est détecté dans une certaine limite (distance ou angle). Vous pouvez spécifier une propriété que l'objet détecté doit avoir.



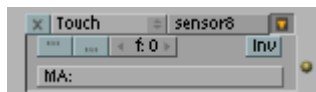
Near - se déclenche quand un objet est détecté à une certaine distance. Vous pouvez spécifier un propriété que l'objet détecté doit avoir.



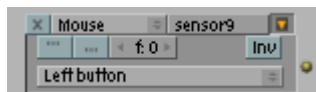
Collision - se déclenche quand l'objet est en collision avec un autre objet. Vous pouvez spécifier un matériau ou une propriété que l'objet heurté doit avoir.



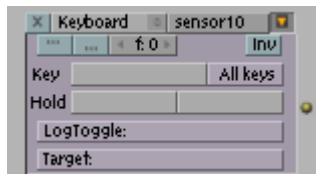
Touch - se déclenche si un objet est touché par un autre objet. Vous pouvez spécifier un matériau ou une propriété que l'objet touché doit avoir.



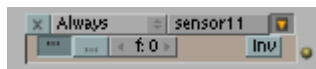
Mouse - se déclenche quand un certain événement de la souris se produit, comme un click de bouton de souris, un mouvement de souris, etc.



Keyboard - se déclenche quand une certaine touche est pressée.



Always - se déclenche à chaque frame.



Controllers: les controllers sont déclenchés par leurs sensors reliés.

AND - lance l'actuator connecté si TOUS les sensors connectés sont déclenchés.



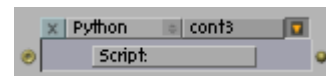
OR - lance l'actuator connecté si l'un des sensors connectés est déclenché.



Expression - évalue une expression.

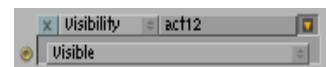


Python - lance un script python.

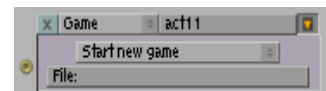


Actuators: si les sensors en rapport sont déclenchés, le controller va appeler le(s) actuator(s) connecté(s).

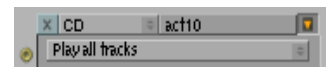
Visibility - Montre et cache l'objet courant.



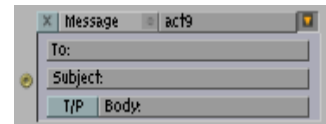
Game - redémarre et quitte le niveau courant. Peut aussi charger une nouvelle scène.



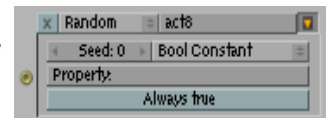
CD - permet de contrôler les pistes sur un CD de musique.



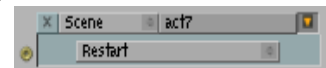
Message - envoie un message à tous les objets, ou à un objet particulier. Ce message va déclencher le sensor Message.



Random - donne une valeur aléatoire dans une propriété de l'objet.



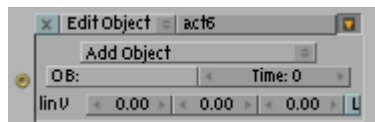
Scene - permet le contrôle sur les scènes - chargement, lancement, suspension, etc.



Ceci est très utile pour montrer différentes scènes, comme une scène de démarrage ou un menu. Quand l'utilisateur veut jouer le jeu en lui-même, un sensor keyboard (par ex. presser [Espace] pour jouer) peut être connecté à un sensor scène, qui pourra alors charger la scène du jeu.

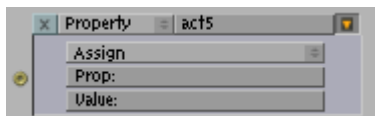
Cet actuator vous permet de spécifier depuis quelle caméra on voit, dans une scène 3D.

Edit Object - permet de contrôler l'ajout, l'édition et la suppression d'objets dans la scène en temps réel. Cela peut servir pour tirer des balles d'une arme.

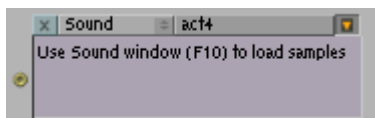


C'est aussi une particularité d'un tracking d'objet.

Property - paramètre la valeur de la propriété de l'objet (ou d'un autre objet).



Sound - vous permet de contrôler des sons dans Blender.

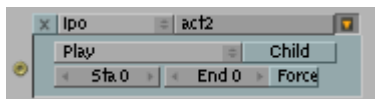


Seuls les sons qui ont été chargés dans Blender seront accessible.

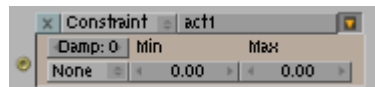
Camera - permet à la caméra de suivre un objet. La caméra peut être mise derrière l'objet (sur l'un des axes X ou Y) et forcée de rester dans une certaine limite (min et max) et à une certaine hauteur.



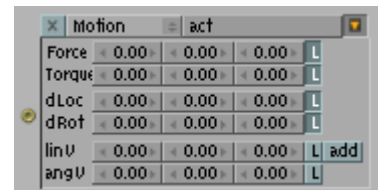
IPO - permet le contrôle sur le lancement d'animation d'objet.



Constraint - contraind la position de l'objet.



Motion - permet le contrôle sur le mouvement de l'objet. Cela inclut le positionnement direct et la rotation de l'objet (dLoc et dRot) aussi bien que l'application de forces à un objet physique pour le déplacer (Force et Torque).



Continuer votre Blender GE

J'espère que vous avez apprécié apprendre les concepts de base du GE. J'espère que vous continuerez à travailler avec, à utiliser les connaissances de base que vous avez apprises ici, à les accroître avec beaucoup de pratique, en apprenant et en échangeant avec les membres de la grande communauté du GE de Blender.

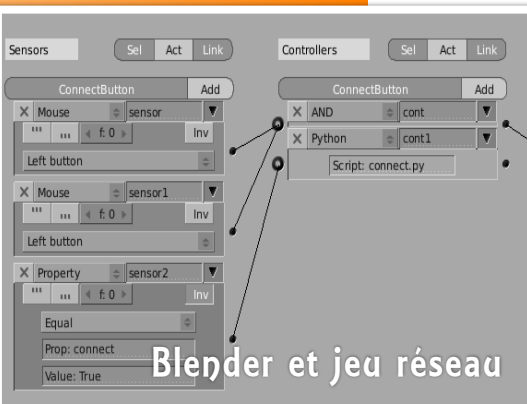
Continuer votre Blender GE

Forum de la communauté Blender Artists - section GE

C'est une des meilleures ressources pour les utilisateurs du GE.

Si vous souhaitez poser toutes les question sur comment faire quelque chose avec le GE, postez quelques exemples de votre jeu actuel, ou simplement vous tenir au courant au sujet du GE, c'est L'endroit principal à visiter. •

<http://blenderartists.org/forum/forumdisplay.php?f=34>



Introduction

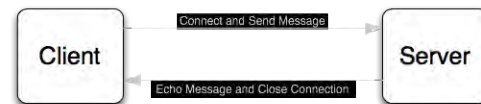
Dans ce tutoriel, nous allons créer une connexion client/serveur pour l'utiliser dans le moteur de jeu de Blender (Blender Game Engine). Vous serez initiés à la gestion bas niveau de réseau en Python à travers l'utilisation des sockets. Nous découvrirons cela au fur et à mesure et avec des illustrations, et l'intégralité des fichiers de ce tutoriel sont disponibles au téléchargement.

Partie 1 : Réseau et Scripting

Prenons donc quelques instants pour expliquer certains points sur le réseau. Je pourrais écrire une encyclopédie complète sur les réseaux informatiques, mais je ne pense pas que vous ayez besoin de connaître davantage de choses que les bases sur la notion de paquet. Il n'est pas nécessaire que vous compreniez parfaitement cette prochaine partie, mais elle est faite de façon à ce que vous puissiez y parvenir. Quand un ordinateur doit envoyer des données à un autre ordinateur, il envoie un paquet d'informations à travers un moyen physique jusqu'à la fin de la réception. Ce paquet est encapsulé avec les informations d'en-tête qui disent où le paquet doit aller. Quand le paquet est envoyé et reçu, il passe par un standard appelé le modèle OSI. La seule chose que vous devez savoir du standard OSI (et du standard TCP/IP) est qu'il fonctionne. Les paquets sont envoyés par un service qui est associé à un port. Un port est simplement un numéro que TCP utilise pour envoyer et recevoir des informations.

Maintenant que j'ai récapitulé environ six mois d'école de réseau dans un paragraphe, regardons le diagramme suivant:

Il s'agit d'un simple diagramme montrant comment nous allons mettre en place notre serveur et notre client. Le client se connectera au serveur et un port de connexion sera établi. Le client enverra une série de données au serveur. Le serveur lira alors les données, les renverra directement au client et fermera la connexion. Voilà. On connaît ce modèle sous le nom de ser-



veur Écho, puisque sa seule fonction est de répéter les données au client.

D'abord, lancer Blender. Pour éviter d'y aller pas à pas avec la création d'un menu de jeu, j'ai pris la liberté d'en créer un à des fins de démonstration. Vous pouvez télécharger le fichier à partir du lien donné à la fin de l'article.

Jetons un oeil à ce que nous avons ici. En bref, il y a une caméra, 3 champs de saisie (le serveur IP, le port et le message), avec un bouton de connexion. Il y a aussi un champ pour la réponse du serveur et un empty que nous utiliserons pour le contrôle de l'interface utilisateur. Commençons par regarder tout cela ensemble.

Pour obtenir un peu d'interaction avec le jeu, nous allons avoir besoin que notre curseur soit visible. Nous allons donc écrire un script Python qui va faire cela. Ouvrez l'éditeur de texte et faites un nouveau fichier appelé "onGameLoad.py". Cela me donnera une bonne occasion de vous initier au scripting en Python, ainsi le reste du tutoriel ne semblera plus aussi intimidant.

Avec Python, nous pouvons ajouter des fonctionnalités presque illimitées à nos jeux. Les 'logic bricks' sont un moyen agréable d'ajouter des fonctionnalités simples sans pour autant connaître de code, mais ils peuvent devenir très complexes lorsque l'on essaie d'ajouter des fonctionnalités qui demandent plus de 20 connexions entre les 'bricks'. En utilisant Python, nous pouvons éviter de trier une toile de connexions de 'bricks' et aller au-delà des limites que les 'logic bricks' nous imposent. Cependant, sans les 'logic bricks', nous n'aurions aucune façon d'appeler nos scripts Python.

Revenons au script "onGameLoad.py". Nous voulons faire un script qui permettra au curseur d'être visible dans le jeu. Le module Python qui gère cela s'appelle Rasterizer.

Pour accéder aux fonctions de Rasterizer, nous devons d'abord importer le module dans notre script. Une fois le module importé, vous aurez accès à toutes ses fonctions. Vous pouvez importer le script en utilisant "import [module]". Pour économiser la frappe, nous utiliserons l'option "as [name]". Cela est très utile lorsque l'on crée de très longs scripts qui nécessitent de taper plusieurs fois le nom du module.

```
import Rasterizer as r
```

Pour afficher la souris à l'écran, utiliser la fonction `showMouse()` comme ceci:

```
r.showMouse(1)
```

Attention : pour appeler la fonction `showMouse()`, nous devons d'abord importer le module. Pour appeler les fonctions contenues dans un module, il faut les référencer avec l'instruction "module.function().subfunction()"

Une autre chose à prendre en compte est l'attribut de la fonction. Dans ce cas, j'utilise "1" pour faire référence à la valeur 'True'. 'False' est représenté par "0". En fait, n'importe quelle valeur non nulle est une valeur 'True'. Nous aurions aussi pu utiliser "r.showMouse(True)" et ça aurait marché de la même façon. C'est une question de préférence personnelle, mais je pense qu'en tapant "1" et "0" c'est beaucoup plus rapide et plus facile que 'True' et 'False'.

Maintenant que notre script est fait, nous devons créer des 'logic bricks' qui vont l'appeler. Nous allons les assigner à notre Empty, appelé "GameController". Sélectionnez l'Empty et allez dans la fenêtre des Logic Bricks [F4]. Ajoutez un Sensor 'Always' et désactivez le 'True level pulsing'. Ajoutez alors un Controller 'Python'. C'est ici que l'on entre le nom du script à charger. Connectez les points, et testez le jeu. Vous devriez

maintenant être capable de voir votre curseur dans la vue du jeu.

Cela devrait vous donner une bonne compréhension sur la façon dont les scripts Python sont utilisés dans le moteur de jeu. Sortons maintenant du mode débutant et intéressons-nous un peu à du vrai code. Je ne vais pas tout expliquer comme "cliquez sur ceci", "mettez vous là", etc. Si vous envisagez sérieusement d'ajouter une gestion de réseau dans vos jeux, vous devriez déjà avoir une bonne connaissance des bases.

Partie 2 : Configuration du serveur Python

Maintenant que l'on a achevé notre cours pour débutant (peut-être une bagatelle pour certains), c'est parti pour la mise en fonctionnement de notre serveur TCP de base en Python. J'aime utiliser l'éditeur de texte de Blender pour écrire mes scripts, surtout parce que j'ai horreur de permuter entre les applications. [CTRL+W] sauvegardera le fichier .blend et [ALT+S] sauvegardera le fichier texte à l'extérieur du .blend.

Ce prochain script devra être sauvegardé à l'extérieur du fichier .blend, parce qu'il sera notre application de serveur. J'ai essayé de trouver un moyen pour lancer le serveur facilement à partir du jeu même, mais je n'ai trouvé aucune manière de le faire simplement. L'exécuter en dehors de Blender permettra de rendre plus rapide les réponses et le traitement des connexions du client.

Jetons maintenant un oeil au coeur des relations réseaux : le serveur. Comme je l'ai indiqué précédemment, nous allons utiliser les sockets de Python pour établir la connexion entre le client et le serveur. Je vais vous montrer le serveur TCP de Python que nous utiliserons et dans le code je le décompose-rai en parties compréhensibles en utilisant des commentaires.

```
# Importons le module de socket
import socket

# Assigne un numéro de port à une variable. Assigner des valeurs
# comme cela rend la modification plus facile, au lieu d'avoir à
# modifier toutes les références au port dans le script. Remarque:
# utilisez un numéro de port supérieur à 1024. Les services les plus
# connus tel que ftp, http, etc. utilisent les numéros de port les plus
# faibles et cela va permettre d'éviter les conflits.
port = 2020

# Ici, nous allons créer notre socket. Nous assignons le socket à 's' en
# appelant le module de socket et la fonction socket(). Les attributs
# de socket() disent au socket de créer un socket de type COURANT
# (en anglais : STREAM) depuis la famille de socket INET
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

# Ensuite, Nous lions le port au hostname. Cela indiquera au serveur
# que nous voulons utiliser un port spécifique pour l'écoute.
s.bind((socket.gethostname(), port))

# Indiquons maintenant au serveur de commencer l'écoute des
# connexions clients. Pour les tests, nous configurons le serveur
# pour traiter 1 connexion simultanée.
# Vous pourrez changer cela par la suite si besoin.
s.listen(1)

# Puisque ce script sera exécuté à l'extérieur du jeu, de préférence
# dans un terminal nous pouvons utiliser print() pour voir l'exécution
# du serveur. J'ai ajouté la valeur du port dans la commande pour
# afficher le numéro du port.
print "Game server is running on port", port

# Voici notre boucle principale pour traiter les connexions de client
```

```
# entrants. Les connexions sont acceptées et le message est stocké
# dans une variable. L'information de la connexion est affichée dans
# le terminal et le message est renvoyé au client. La connexion se
# ferme ensuite.
```

```
while 1:
    conn, addr = s.accept()
    data = conn.recv(1024)
```

```
# Affichage des infos de connexion des clients
print "connected...", addr, data
conn.send(data)
conn.close()
```

Sauvegarder ce fichier sous le nom "server.py". Si vous avez créé le fichier dans Blender, utilisez ALT+S pour le sauvegarder en dehors du .blend. Ouvrez ensuite un terminal, allez dans le répertoire où vous avez sauvegarder le script serveur, et tapez "python server.py" pour lancer le serveur. Vous devriez avoir quelque chose comme ça:

```
$ python server.py
Game server is running on port 2020
```

Pour tuer correctement le processus, utilisez la commande "top" et recherchez "Python" dans la liste, sous la commande. Mémoisez le PID situé à côté et appuyez sur la touche Q pour quitter "top". Ensuite, utilisez la commande "kill [pid]" en remplaçant [pid] par le PID de la commande Python obtenu grâce à top. Après son exécution, vous remarquerez une commande "Terminated" à la fin de votre serveur.

Partie 3 : Création du client

Maintenant que nous avons un serveur qui marche, il ne va pas faire grand chose si nous ne créons pas nos clients. En utilisant le fichier .blend des articles précédents (ici), configurons quelques paramètres avant de créer le code du client.

D'abord, si vous regardez le menu, vous remarquerez que vous avez 3 entrées : le Serveur IP (input1), le Port (input2) et le Message (input3) qui sont utilisés pour créer la connexion avec le serveur. Actuellement, il n'y a aucun moyen simple d'entrer les données pour des objets textes multiples, nous allons donc devoir faire le script.

Avant que nous le fassions, sélectionnez chaque objet en entrée et créez deux propriétés (properties). La première étant une chaîne de caractères (String) appelée "Text" et la deuxième un booléen (Bool) appelé "edit". Positionnez la propriété edit de input1 à True et les autres à False. Cela nous donnera un point de départ.



Je vais juste résumer le script, au lieu de vous le montrer étape par étape. Nous créons un déclencheur (trigger) qui vérifie si un Sensor (Keyboard Sensor) est positif. S'il l'est, alors nous créons une liste en Python des 3 objets en entrée. Nous exécuterons ensuite une boucle if/elif pour affecter les propriétés

"edit" des entrées appropriées suivant les conditions qui sont vraies. Assez simple.

Voici le script. Enregistrer-le sous "tabController.py" dans l'éditeur de texte de Blender.

```
import GameLogic as gl
cont = gl.getCurrentController()
trigger = cont.getSensor("tabcheck")
if trigger.isPositive():
    # get objects
    input1 = gl.getCurrentScene().getObjectList()["OInput1"] # server ip
    input2 = gl.getCurrentScene().getObjectList()["OInput2"] # port
    number
    input3 = gl.getCurrentScene().getObjectList()["OInput3"] # message
    proplist = [input1, input2, input3]
    # server => port
    if (proplist[0].edit == 1):
        proplist[0].edit = 0
        proplist[1].edit = 1
        proplist[2].edit = 0
    # port => message
    elif (proplist[1].edit == 1):
        proplist[0].edit = 0
        proplist[1].edit = 0
        proplist[2].edit = 1
    # message => server
    elif (proplist[2].edit == 1):
        proplist[0].edit = 1
        proplist[1].edit = 0
        proplist[2].edit = 0
```

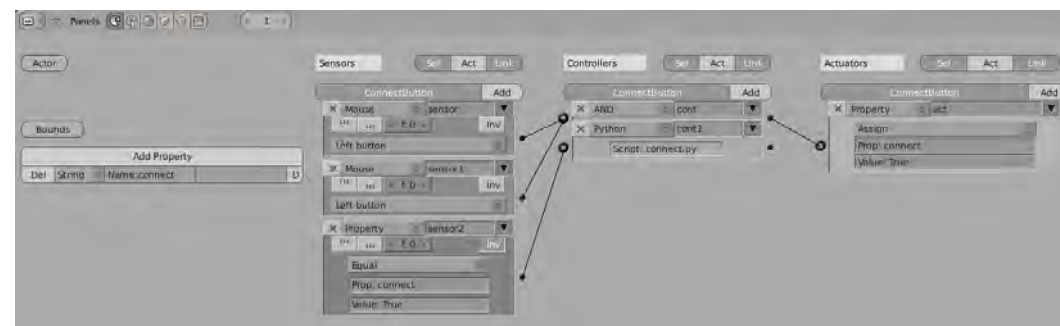
Sélectionnez l'empty « GameController » et ajoutez lui un Sensor Keyboard. Pour cet exercice, je vais utiliser la touche Entrée, parce que si nous utilisons la touche Tab cela retourner des « @ » qui perturbent la saisie de texte. Vous pourriez utiliser la Touche de tabulation si vous voulez, mais vous serez alors obligés d'éditer le script pour enlever le "@" dans l'objet texte. Connectez le Sensor Keyboard à un Controller Python et utilisez le fichier "tabController.py".

va pas marcher parce que chaque Mouse Sensor agira indépendamment comme s'il s'agissait d'une condition OU, et le script s'exécutera chaque fois que le bouton gauche de la souris sera enfoncé et chaque fois que vous passerez la souris au dessus du bouton. Nous devons les connecter à un Controller AND, mais maintenant nous sommes confrontés au problème de ne pas pouvoir exécuter le script directement. Puisqu'il n'y a aucun Actuator Python, nous allons utiliser l'Actuator Property pour mettre une propriété booléenne, qui déclenchera un Sensor Property pour initialiser le script. Regardez l'image ci-dessous pour voir comment cela fonctionne.

Comme pour le script du serveur, je vais vous montrer le script que nous utiliserons et ensuite j'y ajouterai des commentaires pour expliquer ce qui se passe.



C'est ici que l'on voit apparaître un certain nombre de problèmes liés au design des logic bricks. Ce que nous avons à faire consiste à créer deux Mouse Sensors ("Mouse over" et "Left button") et quand les deux sont positifs, le script Python est lancé. Au premier coup d'oeil vous pourriez penser, "Lions les tous les deux à un Controller de type script Python". Cela ne



```
# Depuis que nous utilisons les fonctions du GE,  
# nous devons importer le module GameLogic.  
import socket  
import GameLogic as gl
```

```
# C'est un autre problème avec les logic bricks. Nous pouvons  
# initialiser les scripts en utilisant les contrôleurs Python, mais nous  
# ne pouvons pas les arrêter lorsque leur sensor ne sont pas "true".  
# Si nous ne vérifions pas la propriété, le script tournera deux fois  
# (sur click et sur release)  
# Nous allons créer une boucle pour vérifier la propriété est "true",  
# de sorte qu'elle ne se lance qu'une seule fois.  
conprop = gl.getCurrentController().getOwner()  
if (conprop.connect == True):
```

```
# Maintenant, récupérons les valeurs des propriétés de "Text" des  
# objets inputs et assignons les à leur variables respectives.  
# Notez que nous avons besoin d'utiliser la fonction int() pour les  
# paramètres du port. Sinon, la variable sera assigné comme une  
# chaîne, et vous obtiendrez une erreur lorsque vous connecterez le  
# port car ce ne sera pas une valeur de type integer.  
hostinput = gl.getCurrentScene().getObjectList()["OInput1"]  
host = hostinput.Text  
portinput = gl.getCurrentScene().getObjectList()["OInput2"]  
portstring = portinput.Text  
port = int(portstring)
```

```
# Comme pour le serveur, nous créons le socket.  
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
```

```
# Au lieu d'écouter les connexions, nous nous connecterons au serveur  
s.connect((host, port))
```

```
# Nous intercepterons le message de l'input 3 et nous utiliserons  
# la fonction send() pour l'envoyer au serveur  
message = gl.getCurrentScene().getObjectList()["OInput3"]  
sendme = message.Text  
s.send(sendme)
```

```
# Assignez la réponse du serveur à une variable,  
# comme cela nous pourrons l'utiliser plus tard, et fermez la connexion  
data = s.recv(1024)  
s.close()
```

```
# Maintenant, nous pouvons afficher le message répliqué dans l'objet  
# Text de réponse du serveur en l'assignant à sa propriété Text.  
resp = gl.getCurrentScene().getObjectList()["OResponseValue"]  
resp.Text = data
```

```
# Et finalement, nous réinitialisons la propriété connect à false.  
conprop.connect = 0 Summary
```

Testons le tout. Démarrez le serveur et lancez le jeu. Tapez l'IP du serveur, le numéro du port sur lequel le serveur est actif et un message que vous voudriez envoyer. Appuyez sur Connect et si tout s'est bien passé, le serveur devrait renvoyer votre message, en l'affichant dans la zone de réponse. Vous pouvez voir les connexions s'afficher dans la fenêtre du serveur terminal.

J'espère que cela vous aura donné une bonne compréhension de la façon dont de simples connexions par sockets sont faites et comment vous pouvez obtenir une gestion d'un réseau dans vos jeux.

Dans mon prochain article, nous verrons comment concevoir un simple serveur de chat en Python, en entier, avec authentification par mot de passe, et en réutilisant l'UI de Blender pour créer un client de chat dans le moteur de jeu.

Mais en attendant, si vous avez des questions, commentaires ou problèmes, n'hésitez pas à m'envoyer un email à bchynds@mac.com, ou participer à la conversation sur Blender-Artists dans ce sujet :

<http://blenderartists.org/forum/showthread.php?t=111532> •



Par Rogério Perdiz

Introduction

Pour ceux d'entre vous qui suivent le Magazine BlenderArt, je suis sûr qu'au moins l'un des personnages de l'image ne vous est pas totalement étranger. C'est Orion, qui a été présenté pour la première fois dans l'édition n°4 - Conception de Personnage. Depuis ce numéro jusqu'à aujourd'hui, il travaillait comme l'un des acteurs principaux de mon court métrage de 10 minutes: Orion Tear!

Orion Tear: Orion Tear est mon premier court métrage d'animation. Il a été fait uniquement avec Blender, Gimp, Inkscape, un poste de travail et moi durant 19 mois et 7 jours de travail non stop du matin au soir. Bien que toute la partie graphique du film soit faite, la bande son est toujours en cours de développement par Telmo Cavaleiro, un musicien local et enthousiaste du son EFX, sur son temps libre. La date de sortie est toujours indéterminée, mais tout indique que cela devrait être vers la fin de l'hiver / début du printemps 2008.

Origine: Depuis que mes yeux s'émerveillaient en regardant des vidéos pleines d'action d'un jeu vidéo nommé Final Fantasy 8, de l'ancienne société SquareSoft, j'ai su que c'était le genre de chose que je voulais faire dans ma vie. En sachant cela, avec le temps, j'ai trouvé que dessiner un carré à la règle était extrêmement dur... aussi, j'ai su que je n'étais pas à ma place!

Deux ans après, j'ai trouvé un article dans un magazine de jeux vidéos portugais, décrivant les merveilles d'une application 3D révolutionnaire créée spécialement pour le développement de jeux, appelée Blender 2.0. Ce précieux 1Mo de pur pouvoir de création est arrivé dans un CD, et je me suis dit: "Et bien, je me moque de créer des jeux, mais peu importe; c'est l'occasion de découvrir si je peux faire des films!"

Et devinez quoi? Depuis ce jour, je me suis progressivement amélioré... jusqu'au jour où quelques types ont fait un film appelé Elephants Dream (ED) avec le même logiciel que j'ai appris à tant aimer.



A cette époque, j'avais toujours beaucoup à apprendre de la réalisation d'un film d'animation, mais en raison de l'esprit open source d'ED, il y avait maintenant beaucoup à apprendre des ressources disponibles ... et aussi, comme a dit Aristote, "Ce que nous devons apprendre à faire, nous l'apprenons en le faisant." Ainsi le 04/10/2006 la production de Orion Tear commençait. But principal du projet : Apprendre.

L'article:

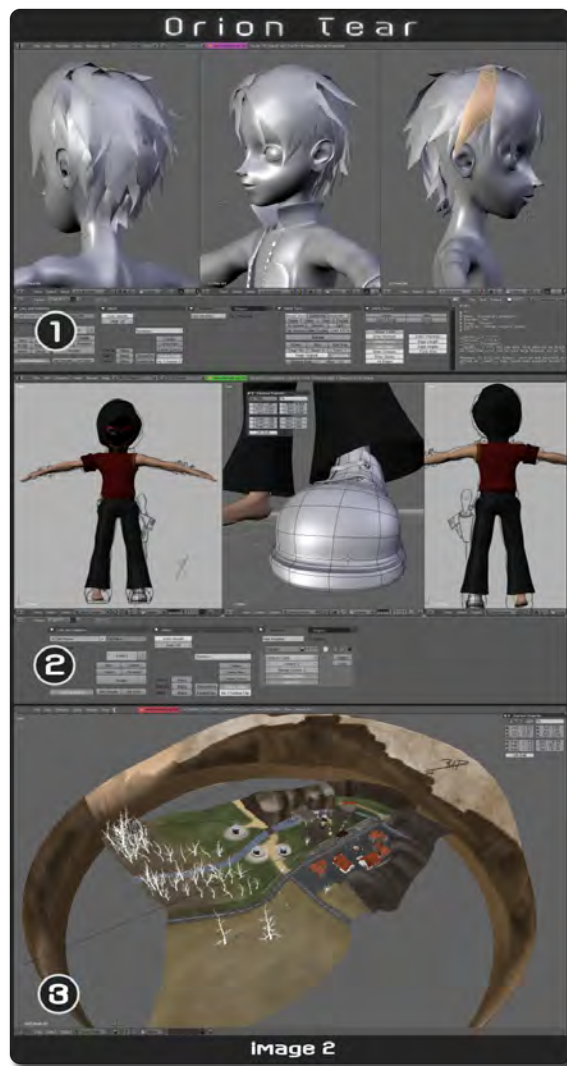
Dans cet article j'essaierai de décrire les étapes accomplies pour l'achèvement de cet exploit Dantesque, mes difficultés, mes pensées, les solutions et toutes les autres choses que j'ai pu approfondir, donc restez à l'écoute... si vous le pouvez!

Esquisses et dessins de conception:

Bien que cela ne semble pas être le cas, cela fut probablement une des tâches les plus difficiles du projet ou du moins une des plus consommatrices de temps. Tout a du être créé à partir de rien.

Pour Orion (img.1-1) j'avais une idée de base. Je voulais qu'il ait les mêmes proportions physiques que Zidane de Final Fantasy 9 et le style de mon dessinateur de personnages préféré (Tetsuya Nomura)... mixé avec mon style personnel de dessin.

Pour Dark, je ne savais pas quoi faire, mais je savais ce que je ne voulais pas... soit un look classique d'ange de la mort. Par chance, au moment de la création, j'ai vu un des westerns de Sergio Leone et c'était parti, un poignard dans une ceinture porte pistolet de western. J'avais à l'époque, et aujourd'hui encore, un téléphone portable avec 2 lumières rouges clignotantes sur le côté et voilà d'où viennent les yeux. La ceinture autour du cou était une erreur. J'ai accidentellement déplacé le calque de la ceinture en haut ... et je me suis dit: "hé! Ça à l'air cool." Le reste est venu naturellement de mon propre style de conception de personnage; j'aime les gants et des vêtements non symétriques.



Les paysages évoluait tout simplement. J'ai commencé à les modéliser directement.

Je voulais quelque chose qui ait l'air naturel avec une subtile pointe de Fantasy. J'ai commencé à partir de l'apparence d'une plage réelle, près de chez moi et l'ai ensuite associée avec des lieux vallonnés. Mais quelque chose manquait, ce n'était tout simplement pas assez accrocheur. Ainsi un week-end j'ai pris ma bicyclette et suis allé faire un tour, sans destination précise, espérant trouver quelque chose pour les améliorer. Je ne sais toujours pas comment, mais je me suis perdu et en essayant de retrouver mon chemin j'ai trouvé un très vieux moulin à vent. D'abord je l'ai presque complètement ignoré et ai continué mon chemin, cherchant toujours quelque chose pour mon paysage, mais, après un moment, j'ai commencé à me dire :

"Mince!!! Je n'ai pas vu ce type de moulins fonctionner depuis des décennies ... c'est drôle d'en trouver un aujourd'hui ... Hum! Un Moulin... Moulin !?"

J'ai immédiatement écrasé les freins et fait un violent arrêt à 180°, j'y suis retourné aussi rapidement que mon vélo me le permettait, j'ai pris une série de photos et après avoir retrouvé mon chemin vers la maison, je l'ai modélisé et maintenant le voici.

Après cela j'ai fait quelques esquisses grossières de la disposition finale du paysage avec des moulins pour thème (img.1-2). J'ai ensuite modélisé une version beaucoup trop détaillée qui s'est avérée être impossible à utiliser... nous en reparlerons dans les chapitres suivants.

Note : Pendant cet article, pour le décor, je me référerai uniquement au décor principal des moulins (img. 2-3). Le film a 3 décors, un dans les montagnes (img. 6-1) et un autre qui est une surprise!

Modélisation des personnages :

Mes compétences en modélisation à cette époque étaient déjà assez bonnes, et, parce que j'avais défini tous les détails pen-

dant la phase conceptuelle, j'ai juste chargé les images dans les vues de Blender et commencé à modéliser.

J'ai commencé Orion par sa tête, ses yeux et ses cheveux; puis le corps et finalement les vêtements. À propos, le modèle d'Orion qui apparaît dans le BAM n° 4 a dû être remodelisé à 80 % en raison des problèmes au rigging, nous en verrons plus dans le chapitre du rigging.

Pour tout ce qui est du corps et des chaussures j'ai utilisé la technique du point par point, c'est-à-dire que j'ai placé tous les points dans l'espace, basés sur les dessins manuels et je les ai alors joint en faisant des faces.

Les vêtements ont été faits initialement par plusieurs Nurbs Circles que j'ai joint en formant des tubes (Malheureusement, j'ai perdu les captures d'écran et il est un peu difficile de l'expliquer). Par exemple : une chemise est seulement un tube subdivisé qui commence d'un côté et se termine de l'autre.

Ensuite en convertissant les tubes nurbs en mesh j'ai pu supprimer les faces dont je n'avais pas besoin. Par exemple, celles par lesquelles passent le corps et le cou. Après quelques réglages et quelques loop cuts, vous obtenez une très bonne topologie de maillage pour la chemise, qui se déforme assez bien.

Le Décor:

Le décor fut mon premier problème. La première version était trop détaillée, avec un tas de polygones inutiles. Je ne connaissais pas vraiment les limites de mon poste de travail, alors j'ai fait des tonnes de détails jusqu'à ce que finalement tout se déplace extrêmement lentement. Je me suis dit : "Et bien, c'est lent mais ça devrait le faire"! Mais ça ne l'a pas fait. J'avais généré 2 millions de polygones. Heureux et content de moi, j'ai appuyé sur le bouton de rendu et devinez quoi? Et bien! 1 heure de temps de rendu. Ce qui ne passe pas très bien pour une animation n'utilisant qu'un seul PC, qui plus est, gêné par l'O.S. Windows qui était limité par 1.5GB de mémoire et avec quelques angles riches en polygones, cela n'a rien rendu du tout.

Atristé par mon ignorance j'ai enlevé tout ce qui n'avait pas d'importance, c'est à dire tout ce qui n'est jamais visible, comme le fond ou l'intérieur des modèles. En les simplifiant au maximum j'ai terminé avec 150 000 polygones (img. 2-3) et un rendu calculé en 5 minutes.

Mais la partie amusante est que le décors high poly n'était pas une perte de temps du tout. J'ai rendu toutes les textures à partir de celui-ci et je les ai appliqué à la modélisation low poly. Nous en verrons plus à ce sujet dans le chapitre suivant...

Texturage:

Bien, le texturage était très amusant! Après que avoir modélisé ce que je voulais, je faisais, en général, tout de suite la texture et seulement après je passais à la modélisation suivante, puis je répétais l'opération.

Je n'ai pas suivi de méthode spécifique en plus de ma méthode, que j'applique encore aujourd'hui. Premièrement, j'unwrap* la modélisation et sauvegarder le calque. Ensuite, je sors et je prends des photos de la chose réelle si elle est facilement disponible, comme par exemple du béton, du bois, du ciel, du tissu, etc. Alors, normalement, je les utilise d'abord comme référence et j'essaye de les reproduire en les peignant dans Gimp ou avec les textures procédurales de Blender.

Ensuite, je sors et je prends des photos de la chose réelle si elle est facilement disponible, comme par exemple du béton, du bois, du ciel, du tissu, etc. Alors, normalement, je les utilise d'abord comme référence et j'essaye de les reproduire en les peignant dans Gimp ou avec les textures procédurales de Blender.

Bien qu'elle fut déjà cool, elle ressemblait surtout à ce qu'elle était, une peinture. Pour qu'elle ait l'air réelle, j'ai pris un petit échantillon carré de la texture de la veste en jean et je l'ai dupliqué jusqu'à ce qu'il recouvre tout le dépliage, en m'assurant en même temps, de ne pas laisser apparaître les coutures.

Après cela j'ai multiplié la peinture par-dessus la texture et le résultat final est ce que vous pouvez voir. Tous les vêtements ont été texturés de cette façon.



A propos des décors :

La plupart des textures des low poly sont des rendus high poly réalisés dans Gimp :) Par ex. : la maison (img. 3-2). A partir du high poly de la maison, j'ai fait le rendu, uniquement avec l'Ambient Occlusion (toujours pas de baking à ce moment :P), d'un mur. Ensuite dans Gimp j'ai ajouté des détails de texture, répétant la même procédure pour tous les murs, le toit, etc. Puis je les ai appliqué sur la modélisation low poly.

Presque tout a été peint dans Gimp, utilisant ensuite l'échantillon de texture comme j'ai fait pour les vêtements.

Une chose qui n'a pas fonctionné aussi bien que je l'attendais furent les montagnes lointaines (img. 2-3). Elles sont faites d'un gigantesque matte painting appliqué sur un tube qui entoure le décor. Si vous vous concentrez sur elles vous noterez que lorsque la caméra se déplace, un très mauvais et non souhaité effet de distorsion apparaît. Après un peu de recherche j'ai appris que c'est un problème connu dans le monde de la 3DCG. De trop grandes textures utilisées de cette façon semblent être propices à causer cet effet.

Beaucoup de personnes me demandent comment j'ai fait le ciel: et bien le ciel est juste constitué de 3 textures procédurales cloud de Blender mélangées. le reste est un arrangement de compositing. J'ai aussi fait beaucoup de tests pour essayer d'avoir un peu de mouvement dans les nuages, mais cela ne fonctionnait pas du tout alors j'ai abandonné et les ai laissés statiques.

La plupart des textures de bois sont procédurales, basées sur celles de la bibliothèque de matériaux de Blender, v 1.01 de Zsolt Stefan Améliorées par moi-même dans Gimp et UV mappées aux objets.

***Dépliage UV (de Blender wiki) :** Pendant le processus de dépliage UV, vous dites à Blender de mapper les faces de votre objet à une image plate dans la fenêtre de l'éditeur d'UV.

Le Rigging:

C'était le plus ennuyeux. Quand j'ai commencé, je ne connaissais rien sur le rigging, mais grâce à Bassam Kurdali j'ai été capable de tout apprendre!



Bassam a fait un personnage entièrement riggé appelé Mancandy et l'a partagé avec tout le monde. En partant de là, à l'époque, c'était le rigging le plus complexe et en même temps le plus facile à utiliser que j'ai jamais vu, ça ne faisait aucun doute... Mais je n'ai pas voulu juste l'importer dans mes personnages. J'ai voulu apprendre, donc j'ai passé 2 semaines à l'étudier et à créer le premier rigging d'Orion. Donc, après 2 semaines, j'avais Orion riggé, mais, je ne savais toujours pas vraiment comment faire un rigging, donc tout était un peu maladroit. C'était suffisant pour les cycles de marche et les mouvements de base, mais je voulais qu'ils soient capables de rivaliser avec Jackie Chan!

Donc j'ai recommencé un autre rigging, cette fois pour Dark, étendant ma recherche aux rigging d'Emo et Proog de ED. Après une autre semaine à démonter les bones, en essayant de tout comprendre, j'ai soudainement commencé à voir toute la logique de la chose et progressivement tout commençait à avoir une signification. 3 autres semaines ont passé jusqu'à ce que j'achève avec succès le rigging de Dark. Après cela je suis revenu à Orion et comme la première topologie de maillage d'Orion était assez mauvaise pour la déformation, j'ai dû presque tout remodeliser et j'ai refait le rigging. Après un mois de plus, il furent complètement prêts pour l'action.

Maintenant, je vais vous décrire la procédure de rigging utilisée pour Orion Tear:

La première étape est valable pour tout type de personnages:

- Nous faisons l'armature des bones (img. 4-1);
- Il est nécessaire de définir quels vertices sont influencés par chaque bone, nous faisons cela en assignant des vertex groups et en utilisant le weight painting pour un réglage précis (img. 4-2);
- Des shape keys correctives sont ajoutées en plus de tout le reste pour assurer une déformation propre, principalement pour des endroits comme les joints et le bassin.

La seconde étape se réfère aux expressions du visage:

- un nombre raisonnable de shape keys sont créés;
- les bones de contrôle sont ajoutés (img. 4-3);
- les shape keys sont paramétrées pour être pilotées par les bones de contrôle.

La troisième étape se réfère à tout ce qui est lié à la dynamique comme les vêtements ou le mouvement des cheveux: Pour les vêtements

j'ai fait une version low poly de la veste, qui supporte la simulation de softbody

- J'ai ajouté des Empties, un pour chaque sommet du mesh softbody et leur ai appliqué un vertex parent à leur sommet correspondant;
- Une autre armature contrainte à l'armature du corps a été nécessaire. Le bout des bones de cette armature coïncide avec l'emplacement des Empties et une lchain IK contrain leur est appliqués.
- Un défecteur softbody est nécessaire (les objets oranges dans img. 4-1) pour éviter l'intersection entre les vêtements et le corps, j'ai fait chaque défecteur parent de leur bone respectif.

Les bones de contrôle sont ajoutés (img. 4-3);

Cela fera suivre aux Empties le mouvement des softbody, les bones suivent les Empties et le mesh des vêtements suit les bones, produisant presque en temps réel le système de simulation des vêtements.

La chevelure est un simple softbody avec un vertex painting pour le contrôle d'influence. C'est la raison pour laquelle ce n'est pas très bon, en fait je n'aime pas du tout les cheveux ... même pas la modélisation, la texture et encore moins le mouvement. Je n'aime tout simplement pas. Mais après avoir passé autant de temps à les réaliser je n'ai pas eu le courage de les refaire... et maintenant des gens me disent qu'ils aiment les cheveux, alors hehe!

L'animation:

De toutes les tâches à accomplir pour la réalisation d'un film d'animation 3D, celle-ci est ma préférée.

J'avais fait quelques animations par le passé, mais aucune avec des personnages riggés. J'ai donc réalisé les scènes par ordre de difficulté d'animation.

J'ai commencé par "les plus faciles" où les personnages se déplacent à peine et j'ai fini par la scène de combat où vous pouvez les voir sauter, se battre, se rouler, tomber, en faisant des cascades etc.

Bien que celle que je pensait être les plus faciles étaient en fait extrêmement difficiles. Juste parce que lorsque quelqu'un ne se déplace pas, cela ne signifie pas qu'ils soit gelé! J'ai dû créer les mouvements de tension, révélant la petite relaxation des muscles que nous avons tendance à faire quand nous sommes à l'arrêt. Le vent dans les manches des vêtements était aussi une key frame animée. La veste d'Orion et la capuche de Dark utilisent une combinaison de simulation dynamique, avec des déflecteurs de type champs de force, et une animation à la main. La plupart du temps, j'ai dû choisir les Empties et les key frame précédemment mentionnés manuellement pour corriger leur mouvement peu réaliste.

La méthode standard alors et toujours utilisée par moi-même est :

- Faites un petit storyboard au brouillon avec les mouvements pour chaque scène;
- Allez à l'extérieur et filmez les mouvements de référence;
- Importez le film de référence dans le séquenceur de Blender et assurez-vous de l'avoir dans une petite fenêtre de prévisualisation;
- En vous basant sur la vidéo de référence, définissez les poses de base;
- Travaillez sur le timing jusqu'à ce que tout semble raisonnablement naturel;
- Effectuez des réglages plus fins sur l'animation;
- Ajoutez les mouvements secondaires comme: les manches, les accessoires, etc.
- Exécutez la simulation pour la dynamique des vêtements et des cheveux;
- Corrigez des dynamiques par les key animant les Empties;
- Finissez par de petits ajustements.

La scène de combat:

La scène la plus difficile de ce film a dû être la scène de combat (img. 5-3). Il y a 2190 frames (1 minute et 28 secondes) d'action continue sans coupures. C'est là où les fonctions d'Animation Non Linéaire (NLA) de Blender sont venues à la rescousse.



Grâce à cela, j'ai pu faire 11 séquences d'actions individuelles (6 pour Dark et 5 pour Orion) et les joindre sans coupure dans les NLA, laissant apparaître le tout comme une seule grosse action. La caméra a été animée en une seule action continue, complétant l'illusion de fluidité.

It took me long time to do, and awfully longer time to render. This one scene uses all the tricks I've learned in all the others previous scenes; more on that in the next chapter...

Rendu/Compositing:

Pour moi, le compositing concerne tout ce qui consiste à convertir les rendus bruts en de magnifiques images avec tous les moyens possibles pour y parvenir. ;)

Je n'ai pas vraiment de technique pré-établie pour ce genre de choses que je fais plus par instinct je suppose. Certaines personnes pourraient appeler ça de la chance... mais cela voudrait dire que j'ai eu de la chance 70 fois (le nombre de scènes)!

Les premières choses requises sont les calques à mixer. Parce que je ne pouvais, ni ne voulais, lancer le rendu d'une traite, j'ai eu, la plupart du temps, 5 calques de base pour travailler: Le décor (Pour simplifier un peu les choses, le set est composé de plus de 10 calques, rendus en un seul), les ombres, l'occlusion ambiante, les personnages et les effets volumétriques.

Tous les calques ont été rendu en OpenEXR, qui permet un meilleur contrôle des couleurs. Puis, les scènes composées ont été re-rendues en Quicktime non-compressé qui a permis une meilleure organisation et utilisation du disque dur

L'éclairage:

La scène est rendue en utilisant l'Ambient Occlusion. Elle est composée de très "peu" de lampes: trois lampes sur les côtés, un Sun coïncident avec un spot pour les halos volumétriques du soleil, une lampe en haut et une autre en bas... plus quelques une le long de la rivière pour essayer de simuler une illumination globale (elles sont plutôt inutiles mais je les ai gardées, je ne sais pas pourquoi).

Les personnages utilisent un dôme de spot* plus le sun. Le centre du dôme coïncide avec la position des personnages et les suivent partout



où ils vont, aboutissant à une apparence d'illumination globale pour un temps de rendu/qualité raisonnable. Aucune Occlusion Ambiante n'est utilisée pour les personnages. Pour quelques scènes j'ai utilisé une technique complexe et très consommatrice de temps pour créer des ombres douces raytracées (img. 6-2). Personne ne l'aurait remarqué si je ne l'avais pas mentionné maintenant. Je l'ai abandonnée dans des scènes suivantes à cause de la viabilité de la chose!

D'abord j'ai fait le rendu de tous les personnages avec le dôme de spots en shadow-buffer puis j'ai rendu la peau et la chemise avec l'ombre raytracée du dôme de spots**, mais tout dans des calques séparées : la Couleur, le spec, les ombres, etc. Puis je les ai rassemblés de nouveau dans le compositeur, mais cette fois-ci avec les ombres floues. Trois ou quatre scènes utilisent cette méthode!

D'ailleurs, j'ai fait deux versions du décors, un pour le rendu et un autre, avec beaucoup moins de détails pour positionner l'animation. Les personnages ne mettent jamais les pieds dans le décor réel. La caméra a dû être importée du fichier des personnages au fichier des décors pour toutes les scènes.

Le plus grand défi a été d'être capable de conserver un éclairage et une ambiance cohérente de scène en scène, car c'est principalement du travail de compositing et il est nécessaire de le faire pour chaque scène. Le décor avait un temps de rendu moyen par image de 5 minutes, et les personnages de 7 minutes. Le calcul des rendus d'Orion Tear a pris environ 5 mois, heureusement, la plupart d'entre eux ont été fait pendant la nuit. Tout a été rendu avec le moteur de rendu interne à Blender.

* Se réfère au mesh d'une demi Icosphere ayant un Spot parenté et l'option Rot de DupliVerts activée, ce qui revient à avoir un spot par sommet se dirigeant vers l'intérieur et vers le centre du dôme.

** Les ombres raytracées donnaient des arêtes franches. La version CVS actuelle de Blender ne pose plus ce problème.

Conclusion:

En fait, l'objectif principal de ce projet a été accompli à 100 %. J'ai appris beaucoup ... mais je n'ai plus vraiment l'envie de faire quelque chose comme cela de nouveau, je veux dire : seull! ... et je ne recommande à personne d'essayer.

C'est juste mon avis, au moment où j'écris ces lignes... Si vous voulez aussi apprendre la fabrication de film d'animation 3D en le réalisant, vous ne devriez alors pas faire plus d'1 minute de film et vous devriez vous concentrer seulement sur une tâche particulière pour chaque film, mais, qui sait...

Je pourrais écrire des centaines de pages sur Orion Tear, mais je suppose que cet article récapitule raisonnablement la plupart des choses, donc c'est la fin de l'aventure avec moi. En espérant que vous aimerez Orion Tear, pour plus de nouvelles gardez un œil sur blenderartist.org!

Links:

Teaser: <http://video.yahoo.com/video/play?vid=338577>

Teaser-b: <http://video.yahoo.com/video/play?vid=562159>

Compositing: <http://videos.sapo.pt/6ZsgTBaZLDZzABOdly4d>

Rigging: <http://videos.sapo.pt/2h21Mr9M5lEoZtzjGzjV>

A propos de moi :

Je suis celui qui a fait Orion Tear, j'utilise Blender depuis la version 2.0 et j'espère être un meilleur réalisateur de film d'animation à l'avenir;

e-mail: rogpertoons@yahoo.com •

MAKING OF: 'Orion Tear'

59

Par Rogério Perdiz





Introduction

J'ai toujours été un grand fan des neveux de Donald depuis que je suis petit. Les nouveaux looks qu'ils avaient dans Couacs en Vrac et leur apparence d'adolescents avec leurs personnalités distinctes les ont faits plus cool que jamais. Et maintenant que j'ai mis la main sur Blender, je ne pouvais pas résister à l'idée de faire l'un d'entre eux en 3D.

Modélisation

J'ai commencé à chercher quelques images de référence du trio sur le Web. Loulou était celui qui, à mon avis, ressemblait le plus à un adolescent. J'ai donc décidé de le modéliser. Je ne pouvais pas trouver parfaitement la correspondance vue face et profils de côté et, pour être honnête, j'étais trop paresseux pour les dessiner moi-même. Alors j'ai fait avec ce que j'avais. La tête est partie d'un cube subdivisé deux fois et converti en sphère utilisant l'outil Sphère du mode Edition (F9). J'ai creusé les orbites en extrudant deux ou trois faces vers l'intérieur. Le bec a été formé à partir d'un simple plan en utilisant une approche de face par face.

Le T-shirt, le pantalon et le chapeau ont été tous modélisés à base de cercles, puis extrudé, en déplaçant et fusionnant les sommets si besoin. J'ai alors ajouté quelques loopcuts [Ctrl+R] au T-shirt et au pantalon pour pouvoir former les plis et les rides.

Pour modéliser les mains, j'ai commencé par les doigts. Ensuite j'ai dupliqué une main, supprimé les doigts et redimensionné un peu pour faire le gant. Je n'avais absolument aucune idée de comment faire ses chaussures. J'ai cherché quelques photos pour avoir une référence. Et j'ai trouvé de belles basket Nike. J'ai donc modélisé l'une

d'elles et je l'ai redimensionné un peu pour lui donner sa forme cartoon qui s'adapte au personnage.

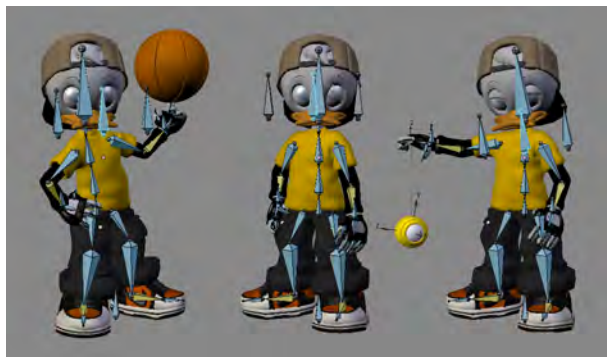


Rigging & Pose

C'est la partie que j'aime le plus, malgré mes modestes compétences en rigging, je dois le dire. Je suis parti pour faire un rigging simple puisque je n'avais pas prévu d'animer ce personnage. Les bras et les jambes sont des chaînes d'IK. Les mains ont été riggées à la même manière de Proog et Emo dans Elephants Dream. J'ai aussi ajouté deux bones en tant que cible pour les yeux.

Les deux os étaient parentés à un troisième qui a été utilisé pour contrôler les yeux. Puisque je n'ai pas eu besoin de couvrir un vaste panel d'expressions du visage, deux ou trois shape keys qui contrôlent la position des sourcils étaient suffisant. Je dois admettre que c'était un de mes premiers rigs complets et il y avait beaucoup d'erreurs qui m'ont posé quelques petits soucis lors de la pose du personnage.

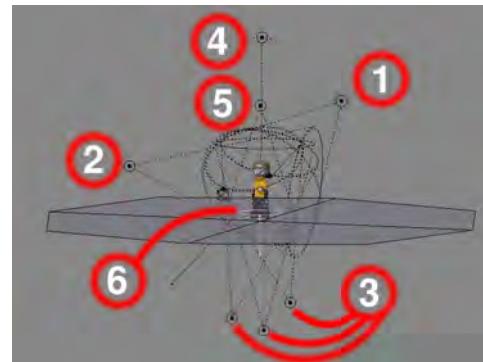
Les yeux avaient leur propre système d'éclairage. Ils ont été modélisés en utilisant la méthode décrite dans le tutorial bien connu "Creating Pixar-Looking Eyes in Blender" (Fig 5). Si vous aimez cette façon de faire les yeux, vous pouvez en abuser. Mais vous devez vraiment porter plus d'attention à l'éclairage. Malheureusement, Blender ne permet pas d'assigner des lumières à différents objets, donc vous devrez juste utiliser différentes couches. J'ai séparé l'iris des yeux (pour le faire, sélectionnez les dans le Mode D'édition, touche P, choisir "Selected") et ensuite placé chaque oeil et chaque Iris sur une couche séparée.



L'éclairage

La configuration de l'éclairage que j'ai utilisé était assez simple. Elle est illustrée dans la Figure 4 :

- 1 Key light (Spot) d'un angle supérieur droit – Buffer shadows activé.
- 2 Fill light (Spot) à gauche, un petit peu plus bas que la Key light.
- 3 Bounce light (Spots). Ceux-là ont été utilisés pour simuler des rayons légers rebondissant du sol vers le personnage.
- 4 Cette lumière Hemi a été utilisée pour éclairer le sol. Il l'avait son niveau d'énergie finalement bien réglé pour que le degré de blancs sur le sol corresponde à la couleur de l'arrière plan.
- 5 Cette lumière (Spot) a été utilisée pour produire la zone orange sur le sol. Bien sûr, le sol était déjà entièrement éclairé par la lumière Hemi, donc une source lumineuse orange normale ne marcherait pas. La seule façon de produire la zone orange était de se débarrasser de la couleur complémentaire (bleue). Donc cette source lumineuse était négative avec une couleur bleue pour soustraire le bleu et laisser l'orange.
- 6 Spot Shadow Only pour produire l'ombre sur le sol. C'était important de montrer que les pieds du personnage avaient un contact ferme avec le sol.



Par Husam Ibrahim

Les yeux avaient leur propre système d'éclairage. Ils ont été modélisés en utilisant la méthode décrite dans le tutoriel bien connu "Creating Pixar-Looking Eyes in Blender" (Fig 5). Si vous aimez cette méthode de faire les yeux comme je fais, vous devez l'utiliser et en abuser. Mais arriver le plus de cela vous devez vraiment porter plus d'attention à l'éclairage. Malheureusement, Blender ne permet pas d'assigner des lumières à différents objets, donc vous devrez juste utiliser différentes couches. J'ai séparé l'iris des yeux (pour le faire, sélectionnez les dans le Mode D'édition, touchez P, choisissez "Selected") et ensuite placé chaque œil et chaque Iris sur une couche séparée.

Chaque œil a été éclairé par une lumière Hemi et avait une lumière spot soigneusement placée pour obtenir un point spéculaire sur la cornée. Chaque Iris a été éclairée par un projecteur brillant depuis un angle afin d'obtenir un gradient coloré agréable à travers l'iris. Cela le faisait avoir l'air vraiment beaucoup mieux et lui donnait une certaine profondeur. Bien sûr, vous pourriez réaliser le même effet en utilisant une texture mappée au canal de la couleur de l'iris avec le mode de mélange Multiply, mais le

contrôle de la direction de l'angle de la lampe n'aurait pas été facile. J'ai finalement parenté les iris et les lumières des yeux pour qu'ils aient la même apparence quelques soit le point qu'ils visaient.

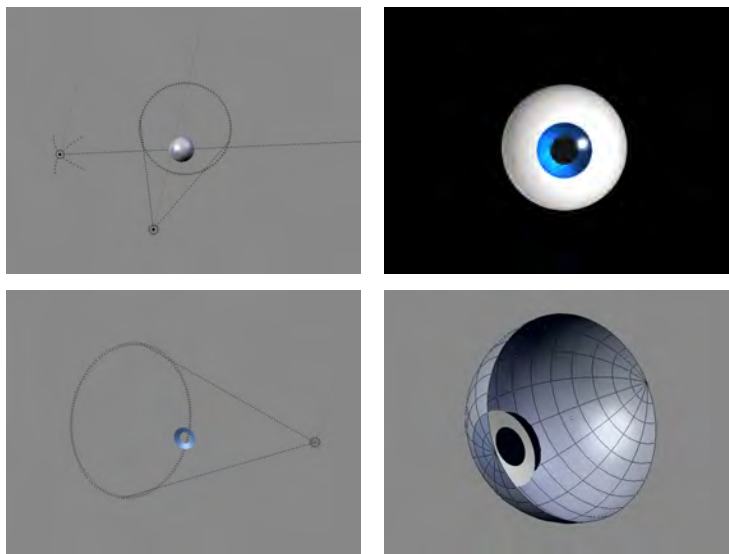
Matériau & Ombres

Il était tentant d'abord de voir à quoi cela ressemblerait avec l'ombrage 3D. Je dois admettre que c'était une mauvaise idée. Parfois vous devez juste laisser les classiques rester classiques, si le cas s'applique. Donc j'ai décidé d'utiliser un Toon shader pour son corps. Ce que je n'aime pas dans la façon avec laquelle les gens utilisent le Toon shader dans Blender est qu'ils aiment avoir des transitions brutales entre des zones éclairées et ombrées.

Ce que les gens ne semblent pas comprendre, c'est qu'ils peuvent obtenir une belle apparence des personnages en faisant exactement le contraire, une transition fortement lissée entre les zones éclairées et ombrées. Cela aide aussi si vous donnez au shader une valeur « Size » moyenne à haute pour faire une zone ombrée la plus petite possible, particulièrement si votre personnage est devant un arrière-plan lumineux.

Les autres ombres que j'ai utilisées étaient principalement Lambert pour le gant et Minnaert pour les vêtements et le bec. C'est vraiment juste une question de réglages pour voir quelle combinaison marche le mieux. Presque tous les matériaux que j'ai utilisés avaient un ramp shader, ce qui donne une apparence agréable, avec un falloff de couleur type pêche. Pour le matériau du corps, le ramp shader n'a pas seulement servi à l'esthétique de l'image, mais il a aussi aidé à séparer le personnage de l'arrière plan, comme s'ils avaient la même couleur.

Les seules textures que j'ai utilisées étaient une texture de jeans que j'ai trouvée sur le Web et le logo de Mighty Ducks que vous voyez sur sa poitrine. Le logo a eu besoin de quelques ajustements de couleurs pour qu'il corresponde avec la couleur du T-shirt.



Par Husam Ibrahim

MAKING OF: 'Teenage Duck'

63

Les deux textures ont été mappées en utilisant une texture basique disponible dans Blender, donc aucun UV mapping n'a été nécessaire. La figure ci-dessous montre les réglages que j'ai utilisés.



Par Husam Ibrahim

Compositing

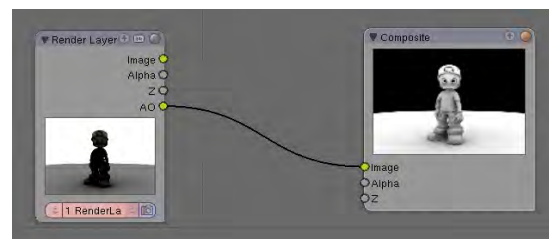
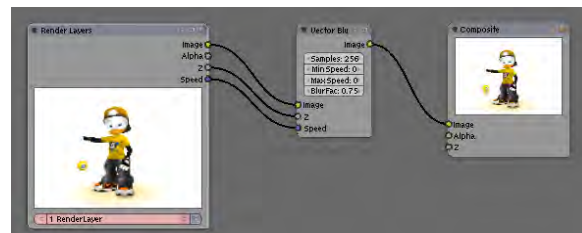
Pas beaucoup de chose ne s'est faite à cette étape, à part une passe d'ambient occlusion que j'ai rendu pour augmenter l'éclairage de l'image et l'effet flou pour le yo-yo et le basket-ball. C'est malheureux que certains grands rendus deviennent plats juste parce qu'ils ont manqué d'un peu d'AO. Même si vous faites un rendu cartoon, un peu d'AO aide toujours, à moins que vous ne fassiez des projets de rendus super stylisés et des rendus type 2D. Mais en mettant juste l'AO actif, le rendu ne s'améliora pas beaucoup, comme il sort multiplié cela a aboutit à un rendu grainé, même avec un nombre maximal d'échantillons.

Il est toujours mieux de faire le rendu avec une passe séparée et le combiner ensuite avec votre image rendue en utilisant l'éditeur de Noeud, ou votre programme de retouche 2D préféré. Pour faire la passe, activez d'abord "l'Ambiant Occlusion" dans les paramètres "World", allez ensuite dans vos paramètres Scène (F10) et assurez-vous que "Ray" et "Do Composite" soient activés.

Une fois que vous avez votre passe d'AO, essayez de le combiner avec votre rendu d'image en utilisant des modes de mélange différents. Pour cette image j'ai utilisé deux couches d'AO, une sur Overlay et l'autre sur Burn. Vous devriez aussi flouter un peu la passe et jouer avec l'opacité pour vous débarrasser du bruit. Parfois la passe AO peut ruiner votre effort d'avoir fait un éclairage spécial pour les yeux de votre personnage, et vous pourriez vouloir effacer cette partie de la passe.

Si vous utilisez les composite nodes, vous devrez utiliser l'ID Mask node pour faire ce travail. Il y a beaucoup d'informations sur ce type de noeud dans la documentation de Blender. J'ai ajouté comme touche finale l'effet motion blur. Pour le faire j'ai utilisé le Vector Blur node de Blender (Fig. 12), après l'animation du yo-yo et du basket-ball. A nouveau, c'est juste une question de configuration et de vitesse d'animation pour voir laquelle des combinaisons marche le mieux.

Voilà, je suppose que nous avons à peu près tout vu. J'espère que ce 'making of' vous sera utile lorsque le temps viendra et que vous vous serez décidés à faire un personnage cartoon.



Bonne chance à tous et bon blend!

Husam Ibrahim •



Introduction

Qu'est-ce que le MGP ? Le MGP (Monkey Game Project) est un projet de jeu Open Source. "Open Source" signifie qu'il est gratuit à télécharger, à jouer et libre de modifications. Alors pourquoi tant de gens ont-ils passé tout ce temps ensemble sur un projet sur lequel il ne seront même pas payés ? Et bien, c'est une bonne question. Tout le MGP a commencé dans l'esprit d'un homme.

MagicMyshu. Myshu était membre d'un forum de développeurs de jeux. Il s'est

intéressé à la 3D, mais n'a pas voulu dépenser 3 000 \$ pour en faire. Finalement, il a découvert Blender. Il démarra donc son travail en 3D, se concentrant surtout sur la modélisation, à cette époque. Myshu rejoint plus tard un forum de Blender nommé "Blender Artists", un endroit où les utilisateurs de Blender peuvent afficher leur travail, et reçoivent de l'aide pour leur projet. Après quelques mois de jeu autour de Blender, Myshu eut une idée.

Myshu décida de faire un petit projet de jeu, dans lequel les nouveaux utilisateurs de Blender pourraient se joindre à lui et apprendre à faire un jeu. Etant lui-même un noob, il décida d'appeler ce projet "Noobs UNITE!"

Un peu dingue, non ? Mais c'était une grande idée ! Tous les Noobs étaient invités, même s'ils ne créaient qu'un post sur le forum.

"Vous êtes invités à me rejoindre ! Pour travailler ensemble et voir ce que nous pouvons trouver avec ce moteur."

"Attention : Le jeu sera plutôt "hasardeux", un esprit ouvert est requis."

"Nous ne passerons pas des heures ou des jours à planifier, nous allons juste commencer à construire notre projet



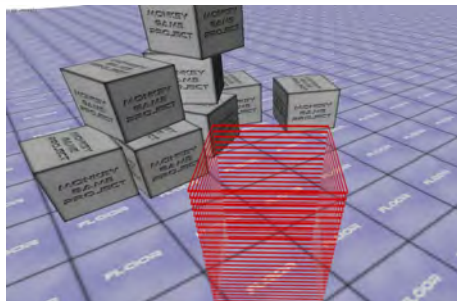
et nous concentrer sur les aspects mécaniques actuels pour apprendre et acquérir de l'expérience."

Les gens ont montré de l'intérêt à rejoindre le projet tout de suite. Dans les 5 heures qui ont suivi, un membre du forum appelé VenomSeven avait modélisé un niveau pour le jeu. Les membres du forum ont continué à montrer leur intérêt pour ce projet "Noob".

Beaucoup plus de gens ont fait de petites contributions. Surtout des modèles, des textures et des sons. Bientôt, Myshu décide de diviser les volontaires en équipes, telles que les décors, la programmation, les personnages et l'animation et les sons/bruitages. Le jeu a progressé doucement. Les membres de la "Noobs UNITE" ont voté pour un jeu de singe (a monkey game en anglais). Le jeu devint alors connu sous le nom de "Monkey Game".

Tout allait bien, jusqu'à ce que le chaos de la gestion de projet rattrape Myshu.

"J'ai envoyé/reçu pas loin de 500 eMails/MP à propos de ce projet aussi bien que j'ai passé environ 30 heures à en parler sur IRC - sans parler des 80 posts que j'ai fait sur le forum également à ce propos - lorsque j'étais malade."

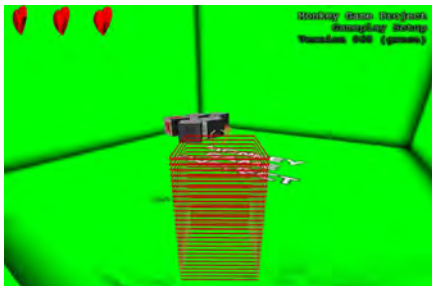


"Monkey Game" a commencé à se disperser. Les choses devenaient silencieuses, les gens s'énervèrent; c'était la folie. Quelques membres ont continué à travailler dur, déterminés à ne pas voir le projet mourir. C'était la période durant laquelle j'ai rejoint le projet. Je connaissais très peu la programmation, mais décidé à prendre un risque, j'ai sauté dans l'équipe de programmation. L'équipe de programmation avait la plupart des éléments nécessaires pour faire le jeu, nous devions juste faire le gameplay.

Lentement, nous avançons petit à petit. J'étais chargé de la vie/santé tandis qu'un autre membre, Stu_Flowers, travaillait sur l'inventaire. Le premier programmeur, Chaser, travaillait sur le gameplay qui était terminé. Mais quelque chose n'allait pas. Nous le savions tous. Monkey Game était mort.

Pendant assez longtemps, Monkey Game resta sur le disque dur du peu de membres qui l'avait conservé, en recherchant des "poussières".

"Personne ne l'a encore dit, mais tout le monde le pense : Monkey Game est mort."



Mais parmi toutes les querelles et hurlements à propos du MGP mort, il y avait encore de l'espoir :

"J'avait sacrifié beaucoup trop de mon temps personnel et de mes efforts juste pour que le MGP meurt..."

Et c'est ainsi que MG (Monkey Game) ressuscita, mais cette fois, en mieux, plus organisé, et plus préparé pour la réalité de la création d'un jeu. Les anciens membres de MG revinrent tous ensemble et commencèrent à travailler dessus à nouveau. Avec de nouvelles voies de communication, il y avait eu beaucoup de progrès.

Au cours des quelques mois suivants, les membres décidèrent de recommencer, et de débiter une nouvelle ère de MGP. MGP avait maintenant un site web qui permettait aux membres de communiquer avec leur communauté, depuis que le forum était rarement mis à jour. Pendant l'été 2007, les membres de MGP firent une pause, en soumettant de temps en temps de nouveaux travaux.

En août 2007, MGP fut publié dans le Blender Game Exposé, ce qui fut une grande preuve de réussite pour les membres.

MGP était retombé sur ses pieds, mais le "chef", Myshu restait introuvable. Nous ne l'avons pas revu depuis. Mais cela n'a pas stoppé MGP.

Comment s'est fait MGP ?

MGP a été fait par une poignée de gens lançant des morceaux de travaux ensemble afin de faire un jeu. Comme je l'ai dit auparavant, MGP était divisé en plusieurs sections. Il y avait "Personnages" et "Animations", qui étaient orchestrées par Myshu.



Ils travaillèrent sur toute la modélisation des personnages, en incluant le texturing, le rigging et l'animation. Une autre équipe était

"Programmation". Nous travaillions sur la mise en place du gameplay et tout autre logic et python nécessaires au jeu. L'équipe "Environnement" travaillait sur l'environnement entier qui était utilisé dans le niveau; ce qui incluait la modélisation des niveaux, mais aussi le texturing. L'équipe Sons a bien sûr travaillé sur les sons. => Il y avait également une équipe Mini-Jeux, ils travaillaient sur les mini-jeux que nous avions prévu d'avoir dans le jeu.

La raison pour laquelle MGP ne s'est pas émietté cette fois-ci est parce que toutes les équipes ont travaillées ensemble. Quand l'équipe de programmation avait besoin de nouvelles animations, l'équipe de Personnages et Animations était là avec leurs blends en attente. De plus, nous avons appris de nos erreurs lorsqu'il est mort la première fois que la communication était la clé. L'équipe de programmation travaillait au coude-à-coude, nous nous envoyions des mails les uns les autres chaque fois que quelqu'un avait une nouvelle idée ou avait fini un script. Le reste des équipes aussi travaillait ensemble et s'assuraient que chaque membre était informé des dernières nouvelles.



Beaucoup de journées ont été consacrées à MGP, près d'un an passé depuis son commencement.

"Nous n'avions pas vraiment de plan spécifique, nous avons donc simplement lancé tous ensemble nos modélisations, textures, dispositions et idées et de cela est apparu un modèle que nous pourrions utiliser pour faire les niveaux"

"Nous avons fait une tonne de contenu, mais 80% de ceux-ci n'ont même pas été mis dans la démo, donc nous projetons de sortir deux versions séparées de MGP."

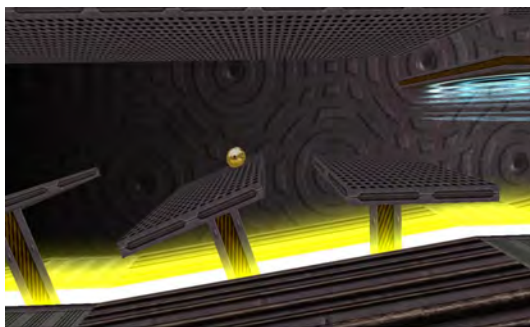


Comme VenomSeven l'a indiqué ci-dessus, il y aura deux versions de la demo du MGP disponible quand nous la sortirons, une version démo régulière qui inclura une démo complète de MGP, la seconde sera une démo Special Edition de MGP qui inclura une démo complète, et les fichiers qui n'étaient dans la démo, et peut-être d'autres choses.

Programmation?

La démo de MGP présentera plus de 700 lignes de code avec des tonnes de briques logiques. La programmation est basée sur un système d'empty, qui possède des empty qui contrôlent les fonctions, apparentées au joueur. Cela donne à l'équipe de programmation un avantage pour éditer une fonction, sans salir aucun des autres. La plupart des scripts python sont expliqués dans le fichier, vous devrez donc juste attendre pour les voir. =>)

MGP a, petit à petit, fini par sortir une démo. Très peu d'entre nous nous ont laissé tomber maintenant, et avec une sortie dans seulement quelques jours, nous espérons que nous pourrions montrer à la communauté combien nous avons appris car c'est à partir de cela que MGP a été fondé. •



ETUDE DE CAS: 'Le moteur temps réel de Blender dans le Boulevard Roosevelt de la SR-686'

68



Visualisation sur Blender RTE

Par Brian Treacy

Introduction

Le moteur temps réel de Blender n'est pas seulement limité aux jeux. Cet article explique comment j'ai mélangé des outils Open Source comme Blender et Gimp avec des outils propriétaires comme Microstation et Photoshop, pour réaliser une balade en temps réel dans une proposition d'étude du Boulevard Roosevelt de la SR686. Il sera présenté au Département des Transports de Floride. Le Boulevard Roosevelt est situé à Tampa, en Floride, à côté de l'aéroport international St. Petersburg Clearwater

Parmi les plans finaux de reconstruction d'une partie du Boulevard Roosevelt, ma société (PB) a conduit une étude pour l'ajout d'une importante voie surélevée. Le but de la présentation était de montrer au client, FDOT, la relation entre la passerelle et une des pistes d'atterrissage pour petits appareils de l'aéroport. Les logiciels utilisés ont été Microstation V8 et Inroads pour le travail de génie civil, Photoshop et Gimp pour les photos aériennes et les textures, et Blender pour la modélisation 3d, l'animation et le temps réel.

C'est une manière réaliste de montrer comment un projet proposé s'intègre dans son environnement -un élément important dans l'approche "Context Sensitive Solutions" du développement de projet.

"le CSS est une approche collaborative et interdisciplinaire qui implique toutes les parties prenantes pour développer des infrastructures de transport qui répondent aux exigences physiques, préservent le cadre, l'esthétique, l'histoire et les ressources environnementales en conservant la sécurité et la mobilité. CSS est une approche qui prend totalement en compte le contexte dans lequel un projet d'amélioration des infrastructures de transport aura lieu. Les principes du CSS incluent la participation précoce, continue et significative

de du public et de toutes les parties prenantes tout au long du processus de développement du projet."

"Qu'est ce que le CSS" FHWA 3 JAN 2008, 09:01:08 GMT.

< <http://www.fhwa.dot.gov/csd/what.cfm> >.

Etape 1 - Réunir les conditions existantes

DTM est l'abréviation de Modélisation Numérique de Terrain (Digital Terrain Modeling). C'est un maillage de surface triangulé montrant la topologie du sol. Les ingénieurs du génie civil les utilisent en tant que plateformes à partir desquelles ils conçoivent les routes et les ponts. Les géomètres professionnels les génèrent typiquement sous forme de contours avec relevés d'élévation. Les ingénieurs les importent alors dans des logiciels de CAO comme GEOPACK, Inroads ou Civil 3D et convertissent les données en triangles. Blender n'a pas de script d'importation de fichiers au format DGN donc j'ai essayé de convertir les 222 842 triangles en DXF mais le fichier est devenu si volumineux que Blender ne pouvait pas les importer. A la place, je les ai converti en 3DS et les ai importé dans Blender.

A ce point, j'ai dû faire face à deux problèmes supplémentaires : le système de coordonnées limité de Blender et 222 842 triangles. Ayant un plan de travail limité, Blender a déplacé arbitrairement mon DTM en dehors de son propre système de coordonnées vers une nouvelle position qu'il pouvait accepter. Avoir 222 842 triangles demande des jours de travail de nettoyage, à enlever les détails superflus pour réduire le nombre de vertices puis convertir la plupart des triangles restants en quads pour les rendre plus faciles à manipuler. Pour régler le premier problème, j'ai sélectionné le contenu entier du DTM et depuis la partie la plus éloignée (d'une valeur de coordonnée XY d'environ 432000, 1295000), je l'ai déplacé vers des coordonnées XY= 0, 0. Je n'ai pas pu résoudre le second problème.

En ce temps (été 2005), Blender ne supportait pas les Normal Maps d'espace tangent, qui auraient repris les détails du DTM dans une texture. Par conséquent, j'ai passé quel-

ETUDE DE CAS: 'Le moteur temps réel de Blender dans le Boulevard Roosevelt de la SR-686'

69

Par Brian Treacy

ques jours à nettoyer manuellement la surface du maillage en concentrant la majorité des détails dans les courbes de la route. Le maillage résultant contenait 171 606 polygones.

Les ingénieurs civils utilisent aussi des "Digital Rectified Orthographic Photos", qui est une longue expression pour désigner des photos aériennes redimensionnables. A nouveau, un topographe vole au dessus du lieu en prenant des photos avec un appareil spécial puis utilise un logiciel (comme Descartes) pour rendre les clichés redimensionnables. Les vues aériennes que nous recevons couvrent 10572x4608 pieds (soit 2700x1200 m), avec une résolution de 6 pieds (1.5m). A cause des limitations de l'OpenGL, j'ai dû découper les clichés en images de 512x512 px, ce qui a donné une grille de 9x21.

Un ingénieur a "drapé" la grille sur le DTM (le drapage est similaire à la fonction de retopologie de Blender, ou les vertices de la grille sont projetés sur le DTM). Cette méthode aligne exactement la photographie aérienne sur la topographie du DTM.



Photo aérienne sur la grille

J'ai alors utilisé l'éditeur UV/Image de Blender pour placer les images aériennes découpées sur les frontières de l'image drapée.

Lorsque l'on travaille sur une zone relativement plate comme Tampa, un DTM est presque inutile pour une présentation. Il n'y a pas assez de variations d'élévation pouvant permettre aux gens de reconnaître où ils sont. Par conséquent, vous utilisez des immeubles pour donner cette impression de familiarité. A la fois les relevés topographiques et les photos aériennes

fournissent les représentations aplaties des immeubles, mais elles ne correspondent pas au réalisme requis pour une pré-



UV map du bâtiment

sentation. Donc nous avons déterminé quelles constructions étaient visibles depuis la route et les avons photographiées. Les clichés ont nécessité un important nettoyage pour que les arbres, lignes électriques et voitures soient remplacés par une peinture virtuelle. J'ai réalisé cela sous Photoshop. J'ai aussi extrudé les contours des immeubles sous Blender pour reproduire les murs et les toits. J'ai utilisé l'éditeur UV/Image pour plaquer les photos aux murs, pour les toits j'ai utilisé les photos aériennes.

Pour obtenir une voûte céleste réaliste, j'ai envoyé un ingénieur sur la route de Tampa avec un appareil photo un jour de beau temps. Elle s'est arrêté sur le pont et a pris une belle (presque complète et sans obstruction) vue panoramique à 360° du ciel. J'ai assemblé les clichés sous Photoshop, puis les ai UVmappés sur un tube dans Blender.

J'ai redimensionné le sommet du tube pour que son rayon soit plus petit que celui de la base, produisant ainsi une forme de cône. Finalement, j'ai créé une animation IPO du ciel avec une légère rotation et lui ai assigné un contrôleur "always" pour que les nuages se déplacent lentement.



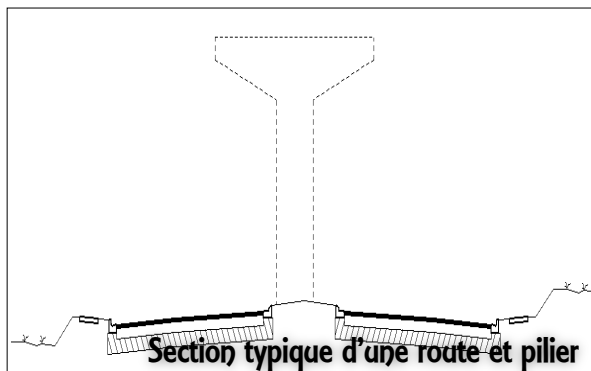
Ciel assemblé

ETUDE DE CAS: 'Le moteur temps réel de Blender dans le Boulebard Roosevelt de la SR-686'

70

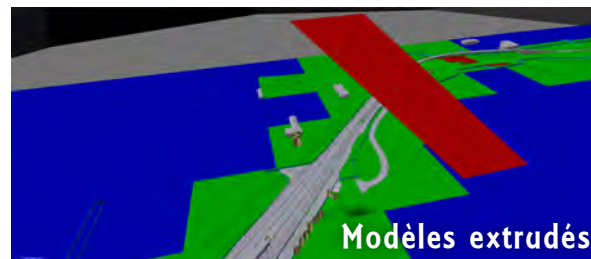
Etape 2 - Construction de l'étude proposée

Pour concevoir une route, les ingénieurs produisent différents types d'éléments : l'alignement horizontal, l'alignement vertical, la section typique, et des sections. L'alignement horizontal est une courbe 2D représentant le tracé que la route, rampe ou pont va suivre. L'alignement vertical (aussi connu sous le nom de profil) a une fonction similaire hormis qu'il fournit les différences d'élévation. La coupe typique donne une représentation générale de ce à quoi une tranche de la route ressemblerait alors que les sections donnent une représentation exacte d'une coupe à un endroit précis. Un projet standard nécessite ceci pour chaque route, rampe et pont.



J'ai combiné les alignements horizontaux et verticaux dans Microstation pour produire un unique alignement 3D et l'ai importé dans Blender. Suivant le même mode opératoire que pour le DTM, j'ai d'abord placé les données dans Microstation pour déterminer les coordonnées; puis sauvegardé sous 3DS avant l'importation. Une fois dans Blender, je les ai tracées en utilisant une courbe de Bézier. J'ai ensuite combiné quelques sections et sections typiques pour former vingt-six segments spécifiques de six routes, rampes et ponts différents. Je les ai importé dans Blender et tracé en

tant que courbes. Finalement, j'ai extrudé les sections typiques le long des courbes pour produire les modèles 3D.



Dans la terminologie des routes, une embranchement est une zone de transition où au moins deux routes convergent ou divergent. A cause des différences d'élévation et du respect des critères de conception, dessiner un embranchement est un défi, surtout quand vous voulez le rendre utilisable par le moteur 3D temps réel de Blender. Pour l'essentiel, j'ai manuellement lissé les maillages là où les rampes rejoignaient les artères et où les routes se croisaient. Rétrospectivement, ce fut sans doute la tâche la plus difficile.

La modélisation du pont pour la passerelle fut très simple et similaire aux autres routes et rampes. J'ai extrudé les sections typiques le long de la courbe 3D pour créer la superstructure. Puis j'ai modélisé manuellement les piles carrées, les voies d'accès et les contreforts sous Blender.



Par Brian Treacy

ETUDE DE CAS: 'Le moteur temps réel de Blender dans le Boulevard Roosevelt de la SR-686'

71

Par Brian Treacy

Etape 3 - Texturage et peinture du projet

J'ai collecté des textures à partir de photos de projet similaires de route. Les textures que je n'ai pas pu trouver, j'ai dû les créer. Je naviguais constamment entre Photoshop et Gimp pour produire les textures. Photoshop pour l'ajustement des couleurs et le redimensionnement et Gimp pour les motifs et la transparence. J'ai aussi créé de nouvelles textures à partir de zéro, en utilisant des filtres et des gref-fons sous les deux applications jusqu'à obtenir le blend "parfait". J'ai Uvmappé tous les modèles, puis peint manuellement les ombres avec le Vertex Painting.

Etape 4 - Ajout des accessoires

Les accessoires aident au réalisme de la présentation. Par conséquent, j'ai modélisé les feux de signalisation, l'éclairage public, les panneaux, les arbres et les bandes blanches. Basés sur les photos prises au voisinage du projet, j'ai essayé de modéliser les accessoires pour qu'ils ressemblent à ceux que l'on trouve dans cette zone. Par exemple, la majorité des arbres sont des palmiers, et les feux de signalisation sont noirs.



En utilisant la technique des panneaux d'affichage pour les arbres, j'ai ajouté un masque alpha à des photos d'arbre et les ai Uvmappés sur deux plans croisés (formant un "X") dans Blender. Pour les bandes blanches, j'ai utilisé une

courbe 3D comme chemin et un "dupliverts" pour copier un plan rectangulaire blanc représentant les lignes pointillées. J'ai extrudé avec "Bev Ob" des plans rectangulaires le long d'autres courbes 3D pour former les lignes continues blanches ou jaunes.



J'ai ajouté une "enceinte" rouge semi transparente représentant les limites de la parcelle (lignes de propriété), et un plan bleu semi transparent sur le couloir aérien de la piste d'aviation adjacente.



Etape 5 - Ajout de l'interaction utilisateur

Premièrement, je voudrais spécialement remercier Randall Rickett, pour avoir partagé son célèbre "walkthrough demo" - Sans son script Python et son rig, réaliser cette présentation n'aurait pas été possible. J'ai importé le script et le rig de visualisation à la première personne de mr. Rickett dans mon modèle. Avec tant de distance à parcourir, les réglages par défaut rendaient le déplacement d'un bout à l'autre du projet trop long. J'ai effectué de petites modifications de valeur des propriétés pour augmenter la vitesse de la

ETUDE DE CAS: 'Le moteur temps réel de Blender dans le Boulevard Roosevelt de la SR-686'

72

visionneuse. Ensuite, j'ai ajouté des collisions à certaines surfaces seulement (au lieu de toutes) pour conserver un bon FPS. Finalement, j'ai ajouté un bouton pour activer ou non la visibilité des limites de la parcelle. Ceci donnait au client la capacité de voir les limites de sa propriété quand il le voulait mais il pouvait les désactiver quand il avait besoin de voir d'autres détails intéressants.

Les besoins de l'animation

Après avoir sauvegardé l'exécutable et l'avoir envoyé avec les DLL au chef de projet, il m'a demandé de produire aussi une animation de deux minutes survolant la zone. Cela s'est avéré plus facile que je ne l'espérais. J'ai sauvegardé une copie de ma présentation et y ai ajouté un chemin avec une caméra parenté.

J'ai aussi eu à modifier quelques textures (principalement celles avec un masque alpha), en y ajoutant un matériau pour que le rendu soit correct. Après plusieurs animations basses résolution testant différents plans de vol, il a choisit sa favorite et j'ai rendu une animation finale de deux minutes exactement. L'UVMapping et le Vertex painting ont réduit le temps de rendu à quelques minutes!

LE MOT DE LA FIN

La construction et le rendu précis d'un projet durant son étude (autorisant ainsi nos clients et le public à voir le projet fini alors que des changements pouvaient encore survenir) a aidé au développement basé sur les principes du CSS. Les demandes de présentations en temps réel sont en augmentation dans l'industrie de la construction, et l'équipement de Blender est adéquat pour répondre à celles-ci.

Des outils utiles pourraient être inclus dans les futures versions de Blender, comme un script d'import/export pour les fichiers Microstation ".dgn", un système de coordonnées

infini (ou même un système de coordonnées flottantes), des unités du monde réel et un support de grandes images haute résolution dans le moteur temps réel. D'autres outils utiles à mes yeux seraient l'utilisation des Normal Maps sur une espace tangent et l'outil de retopologie pour diminuer le nombre de vertices et faire passer les détails du DTM aux textures.

La présentation complète a pris approximativement trois cents heures de travail/homme. Le client l'a beaucoup aimée (la présentation), tout comme le chef de projet (bien qu'il ait moins apprécié les trois cent heures de travail nécessaires pour la réalisation), et Blender s'est avéré un outil compétitif en milieu commercial. •

Brian Treacy

Designer Senior, PB



Par Brian Treacy

Interview avec M. Blender! sur l'utilisation des Services Web dans Blender

Dans cet article, vous apprendrez comment utiliser les services web dans Blender pour réaliser des jeux ou promenades qui interagissent avec les ressources internet. Il convient d'avoir des notions de XML, Python, et PHP.

Ci-dessous, vous lirez mon interview avec M. Blender.

Moi: Bonjour M. Blender!

M. Blender: Salut!

Moi: - Je pense réaliser un jeu ou une promenade qui interagissent avec d'autres joueurs sur internet.

M. Blender: OK, quel est le problème? Vous pouvez le faire avec des scripts python.

Me: Oui, je sais que je peux le faire, mais... je ne sais pas comment!

M. Blender: Est ce que vous connaissez les services web ?

Moi: Les services web ?!

M. Blender: Oui. C'est une base d'échange entre un serveur web et un client.

Moi: Hmm. Pouvez-vous m'en dire plus sur les services web ?

M. Blender: Les services web ont des méthodes que vous écrivez et voyez dans des programmes. Les deux sont appelés par des programmes locaux, mais la différence réside dans le fait que la méthode du programme local fonctionne localement et utilise les

ressources du système local, pendant que les méthodes du service web fonctionnent sur les serveurs et avec les ressources du serveur. Ce n'est pas une définition complète des services web, mais s'en est assez proche.

Moi: Pouvez-vous me donner quelques exemples d'utilisation de service web dans Blender ?

M. Blender: Ok. Par exemple, si vous voulez :

- que les Utilisateurs de votre jeu s'enregistrent sur votre site web et qu'ils se connectent sur le jeu en utilisant leur login et leur mot de passe (vous pouvez voir cela sur la plupart des jeux en ligne)
- Votre jeu vérifie la présence d'une nouvelle version au démarrage et propose la mise à jour à l'utilisateur
- Pour réaliser une présentation d'une maison à vendre en 3D, les visiteurs peuvent discuter avec le propriétaire de la maison et lui poser des questions, ou d'autres exemples.

Moi: Oh, bien. J'imagine que lorsque les utilisateurs doivent s'enregistrer pour donner leur nom et leur mot de passe, je me connecte à une base de données qui contient des informations sur les utilisateurs, mais quel est le rapport entre Blender, Python, le service web et la base de données ?

M. Blender: Regardez l'image ci-dessous; Je pense que cela vous aidera à trouver votre réponse.



Blender! & Services Web

Moi: Ok, Pouvez-vous me donner un exemple étape par étape?

M. Blender: Oui. Faisons un jeu avec un test de vérification de version, étape par étape.

Moi: Ok. :)

M. Blender: Tout d'abord, nous avons besoin d'un jeu. Vous pouvez utiliser un jeu que vous avez réalisé précédemment, faire un jeu simple ou simplement utiliser la scène standard de Blender avec le cube au milieu et continuer.

Moi: Ok, je vais utiliser une nouvelle scène Blender.

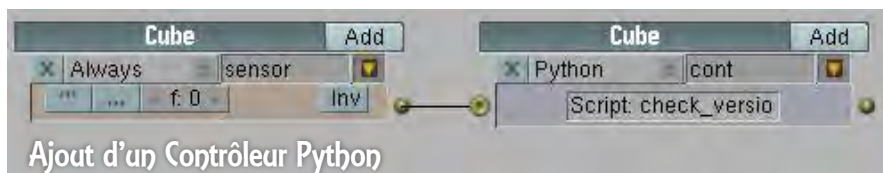
M. Blender: Nous ajoutons en premier un Sensor Always, et décochons Vrai et Faux pour le mode de pulsation.

Moi: Ok.



M. Blender: Maintenant, faites un nouveau fichier dans l'éditeur de texte et appelez le « check_version » sans les guillemets, puis ajoutez un contrôleur Python et configurez le script sur « check_version » sans les guillemets à nouveau, puis connectez le détecteur Always au contrôleur Python.

Moi: Et ensuite?



M. Blender: Avant que vous ne fassiez autre chose sur Blender, nous devons configurer notre service web. J'utilise Apache comme serveur web, PHP pour l'écriture des services web, le protocole XML-RPC pour mon service web, et les implémentations de XML-RPC pour PHP par Keith Deven. Vous pouvez trouver des implémentations XML-RPC pour d'autres langages de programmation sur <http://www.xmlrpc.com/directory/1568/implementations>

Moi: Et ensuite ?

Mr. Blender: Pour les réglages du service web, télécharger en premier l'implémentation de Keith Deven depuis <http://www.keithdevens.com/software/xmlrpc/source>, ensuite installer la sur votre serveur web.

Moi: Ok, Je le télécharge et je l'installe sur <http://en.blender.ir/webservices>.

M. Blender: Ecrivez alors ce code dans une éditeur de texte et sauvez le dans un fichier PHP dans le répertoire où vous avez installé l'implémentation de XML-RPC. Appelons ce fichier server.php.

```
<?php
#1)Include Keith Devens implementation
include("kd_xmlrpc.php");
#2)Web Service Methods
$xmlrpc_methods = array();
$xmlrpc_methods['CurrentVersion'] = 'getCurrentVersion';
$xmlrpc_methods['method_not_found'] =
'XMLRPC_method_not_found';
```

```
#3)Web Service Server
$xmlrpc_request = XMLRPC_parse($HTTP_RAW_POST_DATA);
$methodName =
XMLRPC_getMethodName($xmlrpc_request);
$params = XMLRPC_getParams($xmlrpc_request);

if(!isset($xmlrpc_methods[$methodName])){
    $xmlrpc_methods['method_not_found']($methodName);
}else{
    $xmlrpc_methods[$methodName]($params);
}

#4)Web Service Methods Implementation
function getCurrentVersion($params){
    $returnversion['version'] = 3.60;
    $returnversion['minor']=60;
    $returnversion['major']=3;
    XMLRPC_response(XMLRPC_prepare($returnversion));
}

function XMLRPC_method_not_found($methodName){
    XMLRPC_error("2", "The method you requested,
    '$methodName', was not found.");
}

?>
```

Moi: Je peux le faire, mais je ne sais pas à quoi sert le code.

M. Blender: Ce code initialise un serveur XML-RPC utilisant l'implémentation de Keith Deven, avec deux méthodes, « `getCurrentVersion` » et « `Method_not_found` ». La première méthode est celle que nous voulons appeler à partir de Blender, et la seconde est une méthode pour les requêtes

qui ne sont pas supportées par ce serveur. Dans la première partie du code nous mettons l'implémentation XML_RPC, et dans la seconde section nous configurons un tableau qui sera utilisé avec le code du serveur pour trouver l'implémentation de la méthode de la requête. Par exemple, dans la ligne 2 section 2, nous disons que si une méthode appelée « `CurrentVersion` » existe, la requête du serveur exécutera la méthode d'implémentation « `getCurrentVersion` » de la section 4.

Dans la troisième partie, on fait l'analyse syntaxique de la requête et on extrait le nom de la méthode demandée et ses paramètres. Puis, on vérifie si le nom de la méthode de la requête a été implémenté, et si non, on exécute « `method_not_found` » à la place.

Moi: Oh, Je pense qu'après avoir écrit ce code et l'avoir sauvegardé en `server.php`, nous pouvons envoyer une requête à la méthode et ses paramètres vers `server.php`, et ce code exécutera la méthode demandée et répondra avec des valeurs.

M. Blender: Oui, c'est cela. Une rapide explication, la méthode « `getCurrentVersion` » retourne une structure à trois champs – version, minor et major. Chaque fois que mon programme se met à jour, nous changeons ces chiffres. Dans cette méthode, nous pouvons nous connecter à des bases de données et obtenir certaines informations d'elles.

Moi: Et puis depuis Python, nous envoyons une requête au `server.php` pour exécuter `getCurrentVersion`. Mais comment peut on le faire avec Python?

M. Blender: Cette section est plus facile que la précédente. Python a une bibliothèque appelée « `xmlrpclib`, » et nous pouvons l'utiliser pour envoyer des requêtes et recevoir des réponses. Pour voir ce que peut faire cette bibliothèque, coller ce code dans l'éditeur de texte python de Blender et appuyer sur P.

```
current_version = server.getCurrentVersion()
if(current_version['version'] > this_version):
    print "Go to program web site and download updated
    version."
else:
    print "program is updated."
```

Moi: Je l'ai fait, mais où puis-je voir le résultat?

M. Blender: Vous pouvez voir la ligne s'inscrire dans le terminal/commande prompt de Blender.

Moi: Merci, pour votre temps M. Blender

M. Blender: De rien !

Interviewer - Hamed Zaghaghi,
Etudiant en algorithmes et informatique,
Section Ingénierie en Sciences , Université de Tehran
Email - hamed.zaghaghi@gmail.com •





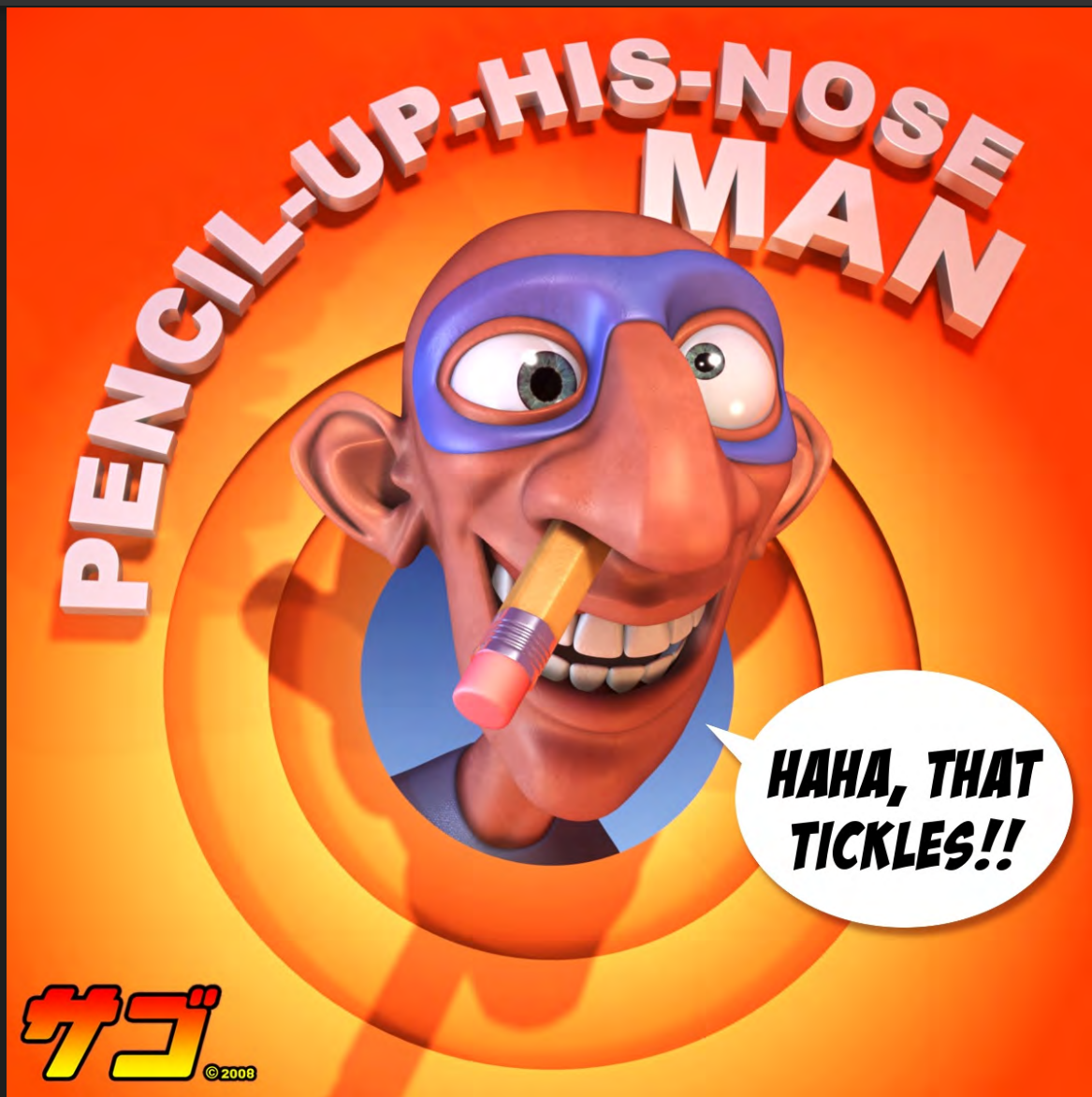






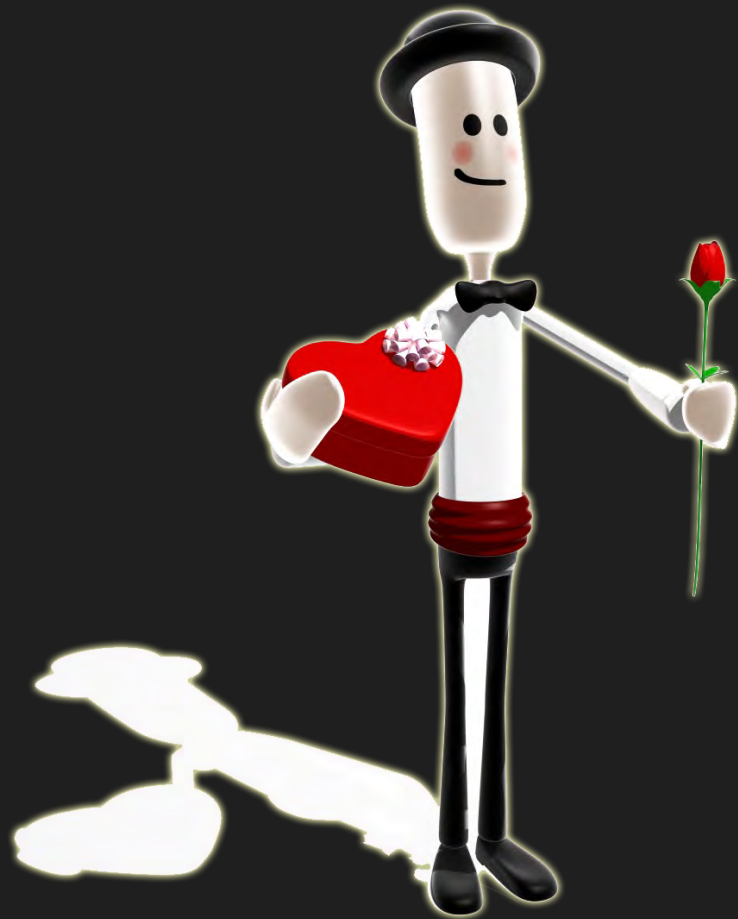






IMAGE COPYRIGHTED © 2007 ROBERT J. TIESS









Voici comment!

1. Nous acceptons :

- Tutoriels explicants les nouvelles fonctionnalités de Blender, les concepts 3D, techniques ou articles basés sur le thème du magazine en cours.
- Reportages sur les événements de Blender à travers le monde.
- Dessins animés liés au monde de Blender.

2. Envoyer vos propositions à sandra@blenderart.org. Envoyez-nous un mail sur ce que vous voulez écrire et nous pourrions faire paraître votre sujet. (Quelques règles à respecter)

- Les images sont préférées en PNG mais des JPG de bonnes qualités feront aussi l'affaire. Les images doivent être jointes séparément du texte.
- Assurez-vous que les captures d'écran sont propres, claires et lisibles et que les rendus sont d'au moins 800px, et 1600px au maximum.
- Les images séquentielles doivent être nommées ainsi : image 001.png... etc.
- Le texte doit être au format, DOC, TXT ou HTML.
- Les fichiers d'archive au format 7zip ou RAR ou moins préféré zip.

3. Merci d'inclure dans votre email les points suivants :

- Nom: Ce peut être votre nom complet ou votre nom d'avatar de blenderartist.
- Photographie: en PNG avec une taille maximum de 256Px. (Uniquement si c'est votre premier article)
- Une petite biographie: 25 mots maximum .
- Site Web: (optionnel)

Note: Toutes les propositions approuvées peuvent être placées dans l'édition finale ou l'édition suivante si elle est considérée convenable. Toutes les propositions seront coupées/modifiées si nécessaire. Pour plus de détails voir le site Web blenderart.

Numéro 15

Animation

- Améliorations récentes des outils d'animation
- Animation faciale: émotions, synchro labiale, phonèmes
- Rendre vos animations plus réelles: arcs, follow through, actions secondaires
- Personnages libre et rigs de personnages
- etc.

Disclaimer

blenderart.org ne prend aucune responsabilité explicite ou implicite concernant la nature ou l'exactitude des informations qui sont publiés dans ce magazine PDF. Tous les articles présentés dans ce magazine PDF ont été reproduit avec la permission exprimée de leurs auteurs/propriétaires respectifs. Blenderart.org et les collaborateurs n'assurent aucune garanties explicites ou implicites en incluant, mais sans limiter à une garantie implicite, l'utilisation marchande ou pour un autre but particulier. Toutes les images et les articles présents dans ce document sont produit/reproduit avec la permission expresse des auteurs/propriétaires.

Ce magazine PDF est archivé et disponible sur le site blenderart.org. Le magazine blenderart est disponible sous la licence Creative Commons 'Attribution-NoDerivs2.5'.

COPYRIGHT © 2007

Les logos 'BlenderArt Magazine', 'blenderart' et BlenderArt sont sous copyright de Gaurav Nawani. 'Izzy' et 'Izzy logo' sont sous copyright de Sandra Gilbert. Tous les produits et noms de sociétés dans cette publication sont des marques ou des marques déposées de leur propriétaires respectifs.

Traduction

Traducteurs du Blenderclan (<http://www.blenderclan.org>)

qui ont participé à ce numéro (par ordre alphabétique):

Atymnia, Aur3D, Batmur, Bjo, EgoN, Flip, Hackira, iet, Lutystrike, Machinegun, Marmouille, Newton, Phil, Teragon, tibo, VincentM.