Rendu Peinture, Interactif et Contrôlable, de Scènes 3D

David Vanderhaeghe, Pascal Barla, Joëlle Thollot, François X. Sillion

Revue Électronique Francophone d'Informatique Graphique, Volume 1, Numéro 1, pp. 53–62, 2007

Rendu Peinture Interactif et Contrôlable de Scènes 3D

David Vanderhaeghe, Pascal Barla, Joëlle Thollot et François X. Sillion

ARTIS GRAVIR/IMAG INRIA Grenoble, France

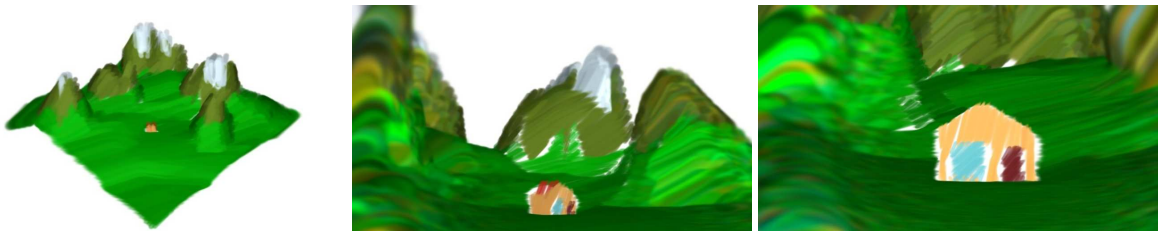


Figure 1: *Trois images d'une animation produite par notre système. Le style utilisé définit la taille des coups de pinceau en fonction de leur distance au centre de l'image (plus fine au centre) et une orientation différente pour chacun des objets.*

Résumé

Nous présentons une nouvelle approche pour créer une animation en rendu peinture à partir d'une scène 3D animée. Notre approche étend et complète les méthodes existantes pour assurer la cohérence temporelle de l'animation, conserver une densité constante de coups de pinceau et offrir à l'utilisateur un libre choix de style. Pour cela nous présentons un algorithme dynamique qui sélectionne à la volée un ensemble adapté de coups de pinceau en s'appuyant sur une hiérarchie de points à la surface des objets 3D. Nous proposons de plus une technique de dessin de coups de pinceau permettant une représentation fidèle de la scène tout en offrant à l'utilisateur la possibilité de spécifier son style via un shader. Le rendu est interactif et effectué sur le GPU.

We present a new approach to create painterly renderings of an animated 3D scene, allowing a controllable trade-off between scene fidelity and stylistic design. Our approach extends and complements existing systems in order to ensure temporal coherence while maintaining consistent stroke density, and allowing a wide variety of stroke styles. It is based on a new dynamic painting algorithm which selects a suitable set of sample stroke locations based on an object-space hierarchy. We describe also an improved stroke drawing technique which allows a faithful depiction of the scene while letting the user specify his preferred stroke style via a shader. All renderings are interactive and performed on the GPU.

Mots clé : rendu expressif, rendu peinture animée, cohérence temporel, contrôle utilisateur

1. Introduction

Le rendu expressif permet d'explorer de nouvelles formes de communication visuelle à l'aide des outils informatiques et de fournir de nouveaux moyens d'expression aux artistes.

La peinture est un médium utilisé depuis la préhistoire. Il permet à une personne (artiste) de communiquer des informations (représentation d'une scène, expression personnelle, idées, sentiments…) au moyen d'une image composée d'un ensemble de coups de pinceau (ou d'un autre outil) sur un support (toile, papier, mur…). La peinture permet, de par sa nature, de faire abstraction de certains détails et d'en accentuer d'autres, ce que nous appellerons *niveaux d'abstraction*. Cette faculté de contrôler le niveau d'abstraction en fait un média puissant pour l'expression et la communication. Ces qualités expressives ont conduit les artistes à l'utiliser non seulement pour créer des tableaux, mais également des animations.

La création d'une peinture animée par les méthodes traditionnelles est un travail très long et fastidieux qui consiste à peindre chacune des images de l'animation. En procédant ainsi il est impossible de contrôler la cohérence temporelle de l'animation obtenue, c'est-à-dire de contrôler la manière dont les coups de pinceau évoluent au cours de l'animation.

Il est donc naturel de penser que les outils informatiques peuvent aider à la production de peintures animées en proposant des systèmes d'édition interactifs et un contrôle de la cohérence temporelle. De plus il est important qu'un tel système propose un contrôle du style et des niveaux d'abstraction pour permettre à l'utilisateur de tirer parti des avantages expressifs de la peinture.

Cet article présente un système de synthèse de peintures animées à partir de scènes 3D, permettant un contrôle rapide et intuitif du style et fonctionant en temps interactif. Nous nous intéressons aux styles de peinture où les coups de pinceau sont perçus individuellement (comme certains styles utilisant la peinture à l'huile, l'acrylique ou la gouache) contrairement aux styles où seuls des aplats de couleur sont perçus (comme l'aquarelle).

L'approche que nous prenons pour créer une peinture de synthèse s'inspire du procédé traditionnel : pour chaque image de l'animation nous définissons un ensemble de coups de pinceau caractérisés par leur position et leurs attributs. Les attributs des coups de pinceau sont définis par le style de la peinture. Ce style est défini en fonction de la scène (notamment en fonction des différents objets représentés) et de leurs propriétés dans l'image (distance à la caméra par exemple) suivant les choix de l'utilisateur. Nous faisons évoluer les positions des coups de pinceau de telle sorte qu'ils suivent les objets qu'ils représentent dans la scène.

L'article est organisé comme suit. Dans la section 2 nous présentons les questions soulevées par la synthèse de peinture animée. Puis dans la section 3 nous faisons un tour d'horizon des travaux précédents. Nous présentons nos contributions dans la section 4. Nous continuons dans la section 5 en expliquant notre approche. Les résultats sont présentés en section 6. Enfin nous discutons les problèmes soulevés en section 7 avant de conclure.

2. Style et représentation

Nous nous plaçons dans le cadre de la peinture figurative, où le peintre représente traditionnellement ce qu'il perçoit. Chacun de ses coups de pinceau représente alors un sous ensemble de la scène perçue, et l'ensemble de coups de pinceau est une représentation de la scène dans un style donné. Mais comment cela se passe-t-il dans une animation ? Les méthodes traditionnelles n'étant pas suffisantes pour permettre le contrôle de la cohérence temporelle, nous nous posons la question de ce que les artistes veulent comme cohérence dans leurs animations. Pour la bonne lecture (compréhension) d'une animation il ne faut pas que le style vienne gêner le spectateur avec du bruit et des phénomènes d'apparition/disparition trop abrupts. Le spectateur doit pouvoir appréhender la scène représentée sans se heurter à des difficultés d'interprétation non désirées par l'artiste.

Pour régler ce problème nous pouvons penser à des approches naïves comme peindre les objets, ou fixer les coups de pinceau en espace image. Aucune de ces stratégies ne fonctionne en pratique. Dans le premier cas nous perdons la nature plane de la peinture. Le résultat ressemble plus à une maquette peinte qu'à un peinture sur un canevas plan (toile, papier…). Dans le second cas nous obtenons un effet dit "rideau de douche" et l'animation semble glisser sous les coups de pinceau car il n'existe pas de lien entre les mouvements dans la scène représentée et la représentation qui en est faite avec les coups de pinceau.

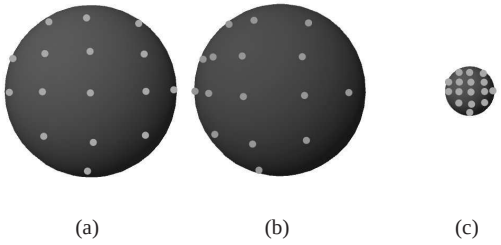


Figure 2: *Changement de densité du au mouvement 3D d'une sphère (a) distribution originelle ; (b) après une rotation ; (c) après un changement d'échelle.*

La question est donc : étant donné un ensemble de coups de pinceau représentant une image, comment peut-on déterminer les coups de pinceau de l'image suivante de l'animation ? Pour assurer la cohérence temporelle, il est naturel de penser qu'il faut retrouver un ensemble de coups de pinceau ayant un lien avec l'image précédente. Il faut donc conserver autant que possible les coups de pinceau d'une image à l'autre et faire évoluer leurs position et attributs en adéquation avec l'évolution de l'animation. Pour cela nous devons choisir comment déplacer les coups de pinceau et comment faire évoluer leurs attributs.

Les méthodes des travaux précédents (voir section 3) consistent généralement à suivre le mouvement des objets de la scène, soit en se basant sur une scène 3D (dans les méthodes basées 3D), soit en suivant le flux optique de l'animation (dans les méthodes basées vidéos). Néanmoins il est possible d'appliquer un autre mouvement aux coups de pinceau, comme dans [HE04], résultant par exemple d'une direction donnée par l'utilisateur.

Le choix de la métaphore de mouvement des coups de pinceau guide les mécanismes à mettre en oeuvre pour la cohérence temporelle des attributs. Dans notre approche où le mouvement des coups de pinceau traduit le mouvement des objets, nous plaçons des points d'attache sur les objets qui servent de points de repère pour le dessin des coups de pinceau (comme proposé par Meier [Mei96]) ; nous devons ensuite faire évoluer les autres attributs suivant les changements dans la scène à représenter, par exemple changer la couleur quand un coup de pinceau passe dans l'ombre. Mais comme les coups de pinceau ne sont pas de simples pix-

els, nous ne pouvons pas directement appliquer la nouvelle couleur car ce changement brutal n'est pas cohérent avec la scène sous-jacente. Il convient donc de correctement adapter les attributs des coups de pinceau au cours de l'animation.

De plus, pour obtenir une bonne représentation de la scène, il faut une densité de coups de pinceau régulière, donc une densité de points d'attache régulière. Or comme les coups de pinceau suivent les objets, nous sommes confrontés à des changements de densité des points d'attache dans l'image (voir figure 2) que nous devons résoudre pour conserver une bonne représentation de la scène. Nous expliquons notre méthode en section 5.2.

Faire l'abstraction d'une scène ne veut pas dire lisser ou enlever tous les détails, mais supprimer certains détails en préservant les éléments clés de la scène comme les discontinuités de couleur, de profondeur ou encore bien représenter les régions et les dégradés. Ces éléments clés sont nécessaires à la bonne interprétation de la scène [Pal99]. Imaginons que l'on peint par dessus une photo, l'utilisation du pinceau supprimera naturellement les petits détails de taille inférieure à celle du pinceau utilisé. Par contre la bonne représentation de la scène est assurée par le fait que l'on respecte les éléments clés : suivre les discontinuités, respecter les dégradés, *etc.* C'est en fait une notion de *fidélité* entre les coups de pinceau utilisés et la scène représentée. Cette fidélité représente la distance qu'il y a entre un coup de pinceau et la scène sous-jacente. Pour intégrer cette fidélité dans le rendu peinture nous pouvons contraindre le dessin des coups de pinceau pour conserver ou non les éléments clés de la scène à représenter comme nous le montrons section 5.4.

3. Etat de l'art

Nous nous intéressons à la synthèse automatique de peinture. Notons toutefois qu'il existe des outils de peinture virtuelle, à l'huile [BWL04] ou à l'encre de chine [CT05], mais ils souffrent des mêmes contraintes que l'utilisation des média traditionnels en ce qui concerne l'animation.

Image fixe Les travaux concernant les images fixes répondent à des problèmes qui surviennent aussi lors d'une animation. Notamment en ce qui concerne la définition et le contrôle du style ainsi que la gestion des niveaux d'abstraction.

Les méthodes de synthèse de peintures fixes prennent en entrée une image 2D ainsi que des informations supplémentaires pour définir le style. Généralement ces méthodes construisent l'image finale en plaçant des coups de pinceau dont la couleur et le style est induite par ces entrées. Haerbeli [Hae90] présente une méthode où des coups de pinceau "simples" sont placés par l'utilisateur en prenant leur couleur à partir d'une image de référence. Hertzmann [Her98] étend cette méthode en construisant, automatiquement, des coups de pinceau plus complexes (un chemin échantillonnant les attributs de style).

Le contrôle des niveaux d'abstraction à été abordé dans les travaux de DeCarlo et Santalla [DS02, SD02] où le niveau d'abstraction est tiré du parcours visuel d'un utilisateur et dans les travaux de Hertzmann [Her01] où le niveau d'abstraction est spécifié par une carte de détails.

Si on applique ces techniques à une animation, bien que le temps de production soit grandement amélioré, il subsiste le problème du contrôle de la cohérence temporelle. En effet, ces méthodes ne sont pas adaptées pour contrôler l'évolution des coups de pinceau au cours de l'animation, cette évolution étant d'autant plus difficile à contrôler si les coups de pinceau sont complexes.

Peinture animée Lorsque l'on fait une animation de peinture il faut se poser la question de la signification que l'on donne au mouvement des coups de pinceau.

Nous pouvons diviser le processus en deux parties, la première définit le mouvement des coups de pinceau alors que la seconde définit la manière dont ils sont dessinés. Ces deux parties sont décorrélées et les techniques de dessin des coups de pinceau d'une méthode peuvent généralement s'appliquer à une autre. Par contre la manière avec laquelle les coups de pinceau évoluent au cours de l'animation dépend d'une part des informations disponibles et d'autre part du choix du style d'animation.

Les techniques de rendu d'animations dans un style peinture se divisent en deux approches. La première utilise une vidéo comme point de départ et ne dispose pas d'information sur la scène pour contrôler le mouvement des coups de pinceau. Introduite par Litwinowicz [Lit97] puis étendue par plusieurs techniques offrant de meilleures propriétés de cohérence temporelle et/ou de style [HP00, WXSC04, HE04, CRH05], ces méthodes ne peuvent pas prendre la scène en compte pour la définition du style car elles utilisent uniquement le flux optique de la vidéo pour définir l'évolution des coups de pinceau au cours de l'animation.

La seconde catégorie, que nous retiendrons ici, utilise une scène 3D animée comme entrée et peut en tirer toutes les informations nécessaires pour le style, comme par exemple le mouvement exact des objets lors de l'animation, les normales des objets ou la distance des objets au point de vue.

Regardons maintenant de plus près les travaux précédents utilisant une scène 3D. Nous avons vu en section 2 que l'utilisation de textures peintes et plaquées sur les objets de la scène ne donnait pas le résultat escompté. Néanmoins Klein *et al.*[KLK*00] proposent une méthode plus poussée utilisant des rip-map, mais n'arrivent pas à contourner l'aspect 3D de leur texture.

Daniels [Dan99] présente une technique où l'utilisateur place des chemins à la surface des objets (espace objet) qui

servent de guides au dessin des coups de pinceau qui se fait purement en 2D (espace image). Il projette l'ensemble de ces chemins sur le plan image et dessine les coups de pinceau correspondants. Bien que les chemins puissent être arbitrairement complexes (un motif par exemple) cette technique demande à l'utilisateur de placer les chemins individuellement sur l'objet et la projection d'un chemin 3D sur le plan image n'est pas contrôlable. Kalnins *et al.*[KMM*02] étendent cette technique en proposant des niveaux de détails, mais qui doivent ici aussi être définis manuellement.

Notre méthode s'inspire de celle de Meier [Mei96] qui consiste à placer des points d'attache à la surface des objets (espace objet) et à dessiner les coups de pinceau à partir de la projection de ces points d'attache (espace image), la définition des chemins se fait alors entièrement en 2D. Haller et Sperl [HS04] étendent cette technique par l'ajout de niveaux de détails. Mais, comme dans la méthode de Meier, il faut judicieusement définir le style des coups de pinceau pour conserver une bonne représentation de la scène. Chi and Lee [CL06] proposent une technique similaire se basant sur QSplat [RL00] pour obtenir les niveaux de détails en segmentant les couleurs des objets pour obtenir des niveaux d'abstraction. Kolliopoulos *et al.* [KWH06] présentent une méthode basée sur une segmentation de tampons d'attributs géométriques avec des coups de pinceau complexes mais le calcul des chemins n'est pas stable au cours de l'animation. Nous proposons une méthode sans contrainte sur le style et offrant des niveaux d'abstraction ne dépendant pas des attributs de la scène. De plus nous effectuons un suivi de l'évolution des attributs au cours du temps et nous offrons le contrôle de celle-ci.

Fidélité En ce qui concerne la conservation d'une bonne représentation de la scène Litwinowicz [Lit97] calcule une carte de discontinuité en utilisant un filtre de Sobel et coupe les coups de pinceau lorsqu'ils croisent une discontinuité. Cette méthode permet de conserver les discontinuités dans la peinture synthétisée mais pas les dégradés.

Kolliopoulos *et al.* [KWH06] proposent de stopper le dessin d'un coup de pinceau lorsqu'il rejoint une zone où la couleur de la scène sous-jacente est trop éloignée de celle du coup de pinceau. En prolongeant la même idée, nous définissons la fidélité comme une distance (définie en section 5.4) entre un coup de pinceau et un point en espace image de la scène à représenter.

4. Contributions

Nous présentons un système complet, s'inspirant des travaux précédents, en améliorant chacune des étapes de la création d'un rendu de peinture animée à partir d'une scène 3D.

Notre principale contribution est de permettre une définition libre du style qui autorise l'utilisateur à lier comme il le souhaite les attributs des coups de pinceau aux données de la scène.

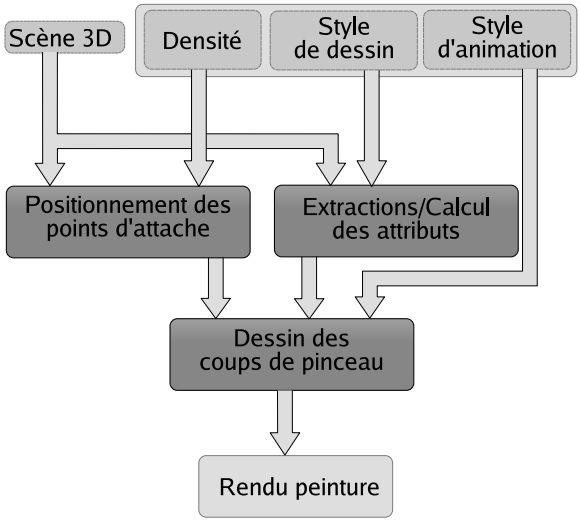


Figure 3: *Pipeline de fonctionnement de notre système : les coups de pinceau sont dessinés à partir de points d'attache et d'attributs déterminés dans les premières étapes du rendu.*

De plus, en offrant un contrôle sur la fidélité de la peinture par rapport à la scène originelle, nous permettons à l'utilisateur de spécifier un style indépendamment de la scène tout en conservant une bonne représentation de celle-ci en modifiant localement les coups de pinceau si nécessaire. Cette notion de fidélité ajoutée à la gestion de la densité des coups de pinceau permet de contrôler les niveaux d'abstraction utilisés.

Finalement nous proposons des mécanismes assurant une cohérence temporelle contrôlable par l'utilisateur.

5. Notre Système

Notre système, présenté en figure 3, prend en entrée une scène 3D animée ainsi que la définition d'un style de dessin et d'un style d'animation fournis par l'utilisateur. Le traitement est effectué ensuite par trois modules. Le premier calcule le nombre et la **position** des coups de pinceau en fonction de la densité souhaitée. Le second extrait les propriétes de la scène à représenter et en déduit les **attributs** des coups de pinceau en fonction du style de dessin. Le dernier effectue le **dessin** des coups de pinceau aux positions déterminées et avec les attributs mis à jour en fonction du style de l'animation.

Nous allons maintenant définir la notion de style utilisée dans notre approche puis expliquer le fonctionnement des différentes étapes.

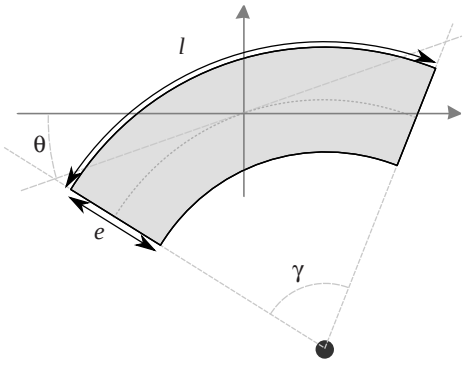


Figure 4: *Attributs géométriques d'un coup de pinceau. Le repère est centré sur le point d'attache. Les distances e et l sont respectivement l'épaisseur et la longueur du coup de pinceau. Les angles θ et γ sont respectivement l'orientation et la courbure.*

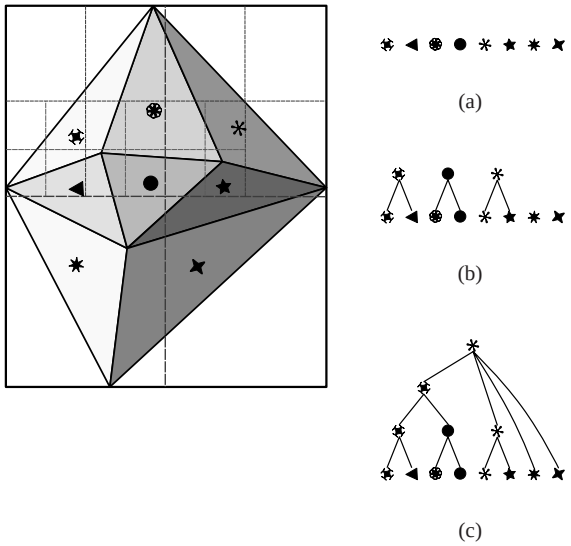


Figure 5: *Construction de la hiérarchie (cas 2D) : à gauche, l'objet plongé dans un quadtree avec une face (son centre) par cellule ; à droite, (a) les centres correspondant aux feuilles de l'octree ; (b) après une itération de la reconstruction ; (c) la hiérarchie correspondant à l'objet.*

5.1. Entrées

Le style d'une peinture décrit comment choisir et dessiner chacun des coups de pinceau qui la composent. Pour pouvoir dessiner un coup de pinceau il faut déterminer ses attributs : couleur et géométrie. Ces attributs sont en lien avec les données de la scène 3D suivant le *style* défini par l'utilisateur.

Dans notre approche, nous prenons en compte les données de la scène suivantes : couleurs, normales, identifiants des objets, profondeur (distance au point de vue) et courbure. Cette liste n'est pas exhaustive et il est facile d'ajouter d'autres attributs au besoin.

Chaque coup de pinceau est défini par une position dans l'image et un ensemble d'attributs : couleur, texture, largeur, épaisseur, orientation et courbure (voir figure 4). Le contrôle des positions se fait avec un paramètre de densité utilisé lors de la sélection des points d'attache, présenté en section 5.2. Le style de dessin définit les liens entre les données de la scène et les attributs d'un coup de pinceau par le biais de choix utilisateur. Cette définition permet à l'utilisateur de fournir une fonction de style utilisant les données de la scène pour chacun des attributs des coups de pinceau (voir les exemples de style en figure 8). Cette approche respecte, suivant l'article d'Hertzmann [Her98], les éléments importants que doit avoir une définition de style pour être utilisable par un artiste, c'est-à-dire des paramètres intuitifs, indépendants de la scène représentée, robustes sur leur espace de définition et indépendants les uns des autres. Nous ajoutons à ce style une valeur de fidélité qui servira lors du dessin des coups de pinceau, voir section 5.4.

Nous pouvons alors calculer les attributs des coups de pinceau pour chaque image de l'animation, mais pour permettre un contrôle de la cohérence temporelle, c'est-à-dire un contrôle de l'évolution de ces attributs, nous devons introduire un autre type de style : le style d'animation. Il détermine comment faire évoluer les attributs d'un coup de pinceau d'une image à l'autre. Pour ceci l'utilisateur donne la fonction qui calcule la valeur de chaque attribut suivant l'ancienne valeur et la valeur demandée. L'utilisateur dispose actuellement de trois styles d'animation : soit prendre la valeur demandée, ce qui produit des sauts dans l'évolution des attributs, soit utiliser une interpolation entre l'ancienne valeur et la valeur demandée, soit faire avancer l'ancienne valeur vers la nouvelle en utilisant un pas fixe. Mais il est facile d'ajouter d'autres fonctions.

Maintenant que nous avons défini les entrées de notre système, nous pouvons choisir les points d'attache qui servent à positionner les coups de pinceau.

5.2. Positionnement

Comme introduit précédemment, nous plaçons des points d'attache sur les objets 3D. Ces points d'attache doivent être placés de telle sorte que leur projection sur le plan image respecte une bonne répartition. Mais lors du mouvement des objets, comme les points d'attache suivent ce mouvement, cette répartition est modifiée et il faut alors mettre à jour les points d'attache pour obtenir à nouveau une bonne répartition tout en limitant les apparitions/disparitions.

Pour cela nous utilisons une hiérarchie de points d'attache construite sur les objets qui permet d'ajouter ou d'enlever

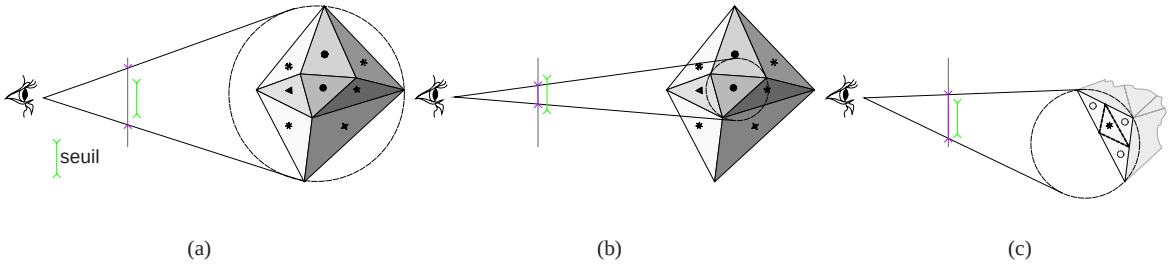


Figure 6: *Sélection des points d'attache : (a) la projection de l'objet est trop grande pour être représentée par un seul coup de pinceau ; (b) on descend donc dans la hiérarchie jusqu'à atteindre un noeud de taille convenable ; (c) si on atteint une feuille de la hiérarchie trop grande, on la subdivise dynamiquement.*

localement des points d'attache en conservant une répartition en espace image relativement bonne. Pour construire cette hiérarchie nous commençons par répartir régulièrement les points d'attache sur la surface des objets de la scène en plongeant chaque objet dans un octree et en plaçant un point d'attache par cellule de l'octree. Nous raffinons l'octree jusqu'à obtenir le centre de chaque face du modèle dans une seule cellule (voir figure 5). Puis, en suivant un parcours de l'octree des feuilles vers la racine, nous construisons une hiérarchie de points d'attache en choisissant pour chaque cellule de l'octree le point d'attache le plus proche de son centre parmi les points d'attache de ses fils. Comme ceci les points d'attache d'un niveau sont présents dans le niveau suivant ce qui réduit les sauts lors d'un changement de niveau dans la hiérarchie.

Enfin, à chaque image de l'animation, nous effectuons une sélection des points d'attache en calculant le rayon projeté de la sphère englobante des faces correspondantes à un noeud de l'octree et en le comparant à un seuil représentant la distance minimum que nous voulons entre deux points d'attache. Si la hiérarchie n'est pas assez dense, nous ajoutons des points d'attache dans la branche concernée en subdivisant la face de l'objet 3D correspondant (voir figure 6).

Une fois que nous avons projeté les positions des coups de pinceau, nous pouvons calculer leurs attributs en fonction du style choisi.

5.3. Extraction/Calcul des attributs

Cette partie consiste à extraire les données de la scène pour la vue courante. Il s'agit en fait de projeter ces données sur le plan image correspondant à la vue courante. Dans la pratique nous rendons des tampons géométriques [ST90] sur le GPU.

Il suffit ensuite de calculer les tampons d'attributs correspondants au style en utilisant les fonctions définies par l'utilisateur. Ces tampons sont lus aux positions de chacun des coups de pinceau composant l'image finale au moment du dessin (voir section 5.4).

L'utilisateur fournit les fonctions sous la forme d'un shader qui prend en entrée l'ensemble des tampons d'attributs de la scène. Ces fonctions peuvent être une simple constante, comme une orientation à 45° ; une fonction de l'espace image, par exemple des largeurs plus fines au centre ; une fonction des attributs, comme la courbure liée à la normale des objets ; ou encore une combinaison de ces possibilités. Pour contrôler ces fonctions par une interface graphique, l'utilisateur peut exposer dans l'interface de dessin autant de variables qu'il le souhaite qui seront utilisées dans le shader. Par exemple s'il veut définir la taille du coup de pinceau de manière interactive, il lui suffit de définir dans le shader que la taille dépend d'une constante "exterieure", le système ajoute alors automatiquement un "slider" à l'interface de visulation permettant de définir cette constante.

5.4. Dessin

Une fois que nous avons les positions des coups de pinceau ainsi que l'ensemble des attributs permettant leur dessin, nous pouvons dessiner les coups de pinceau dans l'image.

Pour chaque coup de pinceau, nous calculons la valeur de chacun de ses attributs en fonction du style d'animation, des valeurs utilisées à l'image précédente et des valeurs demandées (fournies par le calcul des attributs). Un coup de pinceau est donc une courbe paramétrée munie d'une épaisseur (voir figure 4). Pour un paramètre $t \in [0,1]$ le point sur la courbe est défini par :

$$f(t) = \begin{cases} \begin{pmatrix} \frac{l}{\gamma}\sin(\gamma(t-0.5)) \\ \frac{l}{\gamma}(\cos(\gamma(t-0.5))-1) \end{pmatrix} & \text{pour } \gamma \neq 0 \\ \begin{pmatrix} l(t-0.5) \\ 0 \end{pmatrix} = \lim_{\gamma \to 0} f(t) & \text{pour } \gamma = 0 \end{cases}$$

Ici l/γ est le rayon de courbure. En construisant le repère de Frenet de la courbe, nous obtenons le vecteur tangent T et le vecteur normal N :

$$f'(t) = \begin{pmatrix} l\cos(\gamma(t-0.5)) \\ -l\sin(\gamma(t-0.5)) \end{pmatrix} \quad ||f'(t)|| = l$$

$$T(t) = \frac{f'(t)}{||f'(t)||} = \begin{pmatrix} \cos(\gamma(t-0.5)) \\ -\sin(\gamma(t-0.5)) \end{pmatrix}$$

$$N(t) = \begin{pmatrix} \sin(\gamma(t-0.5)) \\ \cos(\gamma(t-0.5)) \end{pmatrix}$$

Pour l'épaisseur, il suffit de déplacer le point suivant la normale à la courbe :

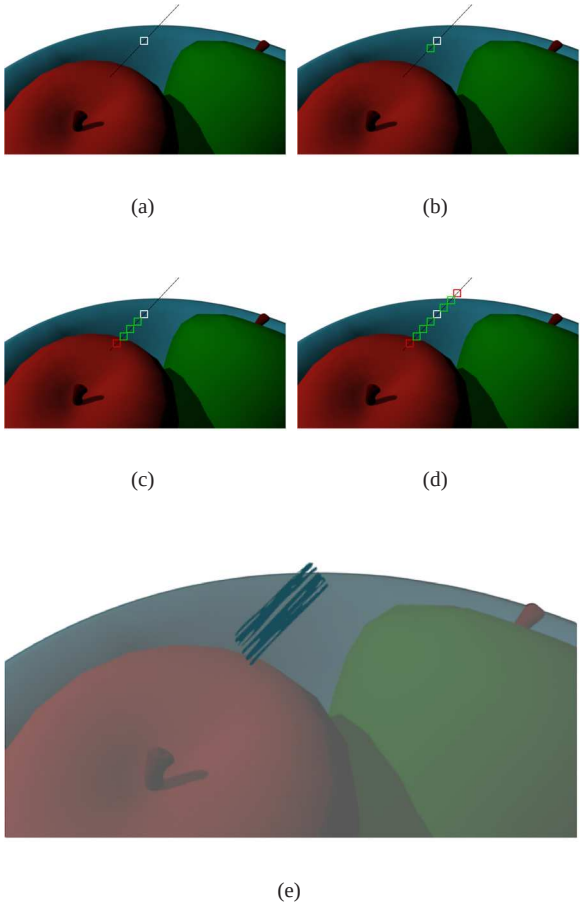
$$f(t) = f(t) \pm \frac{e}{2} N(t)$$

Une fois que nous avons déterminé la courbe représentant un coup de pinceau comme décrit ci-dessus, nous le plaçons en espace image en effectuant la rotation définie par l'angle θ et la translation correspondant au point d'attache.

Lors du dessin du coup de pinceau, nous voulons respecter une cohérence avec la scène représentée, pour cela nous utilisons le paramètre de fidélité. Nous avons choisi d'utiliser la distance euclidienne de la couleur en espace *Lab* combinée (suivant des paramètres utilisateurs) avec la distance euclidienne en profondeur. Nous évaluons cette distance entre les valeurs au point d'attache et les valeurs à la position courante le long du chemin du coup de pinceau. Si cette distance est supérieure aux paramètres de fidélité alors le coup de pinceau est raccourci en conséquence (voir figure 7).

Nous obtenons alors pour chaque coup de pinceau un paramètre maximal d'élongation sur le chemin du coup de pinceau, de part et d'autre du point d'attache. Le dessin des coups de pinceau ainsi obtenus conserve les éléments clés de la scène tels que les changements de couleur et de profondeur, mais permet aussi de bien représenter les dégradés, et ceci sans modifier la définition du style de dessin.

A chaque image de l'animation nous évaluons les deux paramètres d'élongation en chaque position et effectuons la mise à jour en respectant le style d'animation. Cette mise à jour permet de faire des apparitions/disparitions douces lors de l'ajout/suppression de points d'attache.



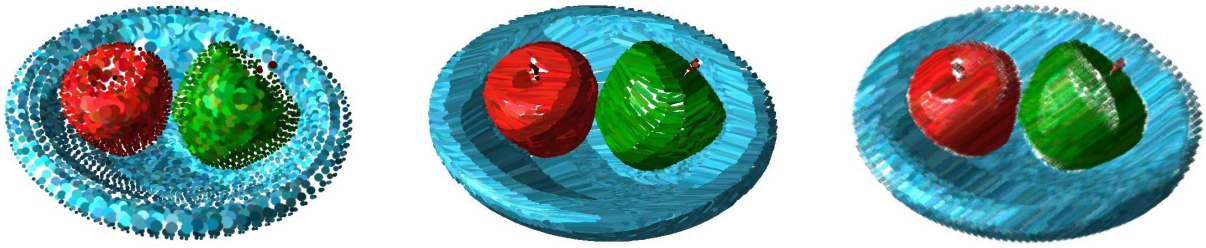
(a) (b) (c) (d) (e)

Figure 7: *Mise en oeuvre de la fidélité. (a) le squelette du coup de pinceau et le point d'attache ; (b) le point considéré respecte la contrainte utilisateur de fidélité ; (c) le point considéré rompt la contrainte de fidélité ; (d) fin du processus ; (e) le coup de pinceau final (image de référence désaturée).*

Puis nous dessinons les coups de pinceau en espace image, par ordre de profondeur décroissante (algorithme du peintre). Nous effectuons un test de visibilité si besoin, par exemple en cas de couverture partielle. En pratique, un coup de pinceau est construit en utilisant une bande de rectangles définie par les attributs géométriques et les valeurs d'élongation. Cette bande peut être texturée.

Pour accentuer l'effet peinture des coups de pinceau, l'utilisateur définit une valeur de perturbation venant modifier par un coefficient aléatoire les attributs de chaque coup de pinceau. Pour ne pas introduire d'incohérence en utilisant cette variation, le coefficient est initialisé à la création du coup de pinceau et reste valide pour toute sa durée de vie.

Dans la pratique nous effectuons le rendu des coups de



(a) (b) (c)

Figure 8: *Différents exemples de style d'une même scène : (a) pointilliste en variant la taille en fonction de l'angle entre la normale et le point de vue ; (b) l'orientation des coups de pinceau est modifiée par la normale des objets pour mieux faire ressortir la forme, la couleur provient d'un toon shading ; (c) l'orientation des coups de pinceau est constante.*

pinceau en deux passes sur le GPU. Lors de la première un point est envoyé à la carte graphique pour chaque point d'attache sélectionné. Nous calculons alors les attributs géométriques et les paramètres d'élongation du coup de pinceau correspondant en utilisant le style de dessin, le style d'animation et la fidélité fournis par l'utilisateur. La deuxième passe trace les bandes de rectangles en fonction du résultat de la première.

6. Résultats

Notre approche (voir figures 1, 8 et 9) permet à un utilisateur de définir un grand nombre de styles de peinture où les coups de pinceau sont perçus individuellement, comme le montre les diverses images présentes dans l'article. De plus la cohérence temporelle est bien assurée par la définition du style d'animation. Les animations (`artis.imag.fr/Publications/2007/VBTS07/`) ne présentent pas de variations brutales d'attributs notables.

La définition d'un style par un graphiste prend quelques minutes, toute la difficulté étant de savoir quel style on veut appliquer à la scène. Les shaders de style utilisés dans ces exemples font entre 15 et 35 lignes.

Le temps de dessin des coups de pinceau (hors sélection) se fait entre 20 et 40 Hz (en remplissant un écran de 800×600) sur une machine équipée d'un P4 3.00GHz et de 2Go de RAM avec une carte graphique GeForce 7800 GT. Cette performance dépend fortement du matériel utilisé, mais de manière générale du nombre de sommets dessinés (points d'attache et ceux composant les bandes de rectangles) ainsi que du nombre de pixels dessinés. Sur les scènes exemples, le nombre de coup de pinceau présents simultanément à l'écran varie entre 200 et 10000.

Pour les scènes que nous avons testées, les structures de données peuvent occuper jusqu'à 400Mo (hors scène 3D) car nous ne faisons que des subdivisions de la hiérarchie de points d'attache. Cette taille varie suivant le nombre d'objets de la scène et suivant le point de vue adopté (elle augmente lors d'un zoom car il faut beaucoup raffiner la hiérarchie). Pour gagner en occupation mémoire il faudrait élager les zones de la hiérarchie qui ne sont plus utilisées. La majorité du temps de calcul est prise par la sélection des points d'attache qui fait chuter le temps de rendu jusqu'à 1Hz dans certaines scènes d'exemples présentées ici lors d'un zoom rapide (c'est-à-dire lorsqu'il y a beaucoup de subdivision d'une frame à l'autre).

Figure 9: *Différents exemples de scène avec le même style que la figure 8(c). On peut noter que le style est transposé d'une scène à l'autre tout en conservant les éléments clés.*

7. Discussions et perspectives

Distribution Notre système de distribution de points est une première solution qui peut être améliorée pour obtenir une meilleure gestion de la densité des points d'attache.

Notre hiérarchie de points d'attache se base sur des maillages statiques, ce qui ne pose pas de problème pour déplacer les objets dans la scène (transformations rigides), mais qui ne permet pas d'utiliser un squelette d'animation ou encore des modèles dont la topologie change au cours de l'animation, comme on le trouve traditionnellement dans les scènes composant un film d'animation.

Nous pensons qu'il reste beaucoup de méthodes à explorer concernant la répartition des points d'attache et notamment celles qui contourneraient ces problèmes en restant indépendantes de la géométrie.

Dessin des coups de pinceau Notre système de dessin de coups de pinceau est relativement simpliste en ce qui concerne les fonctions de mélange. L'amélioration de la qualité du rendu et l'introduction de phénomènes de peinture plus complexes permettraient d'obtenir une apparence plus proche de la peinture traditionnelle. Mais comme cette qualité de rendu a un coût de calcul non négligeable, elle sera réservée pour un rendu hors-ligne, le système conservant un rendu plus simple de pré-visualisation interactive.

Style d'animation et Cohérence temporelle Les fonctions utilisées pour le style d'animation montrent la diversité des comportements que l'on peut obtenir. Dans nos tests nous avons remarqué qu'il est important d'adoucir les transitions sans rendre l'animation trop floue. Hormis cet adoucissement, le choix de fonctions entre dans le style de l'animation et les différences sont plutôt subtiles. Dans notre système la définition du style d'animation se fait pour toute la scène, mais il serait intéressant de l'étendre en permettant de choisir un style d'animation suivant les objets, ou suivant les attributs de la scène, par exemple lier la vitesse d'évolution des attributs à la vitesse de déplacement des objets.

D'une manière générale nous pouvons voir le rendu de peinture animée comme un ensemble de coups de pinceau évoluant au cours du temps. Pour obtenir un contrôle de la cohérence temporelle il faut contraindre cette évolution en effectuant un suivi des attributs. Suivant les contraintes que l'on choisit, le résultat est plus ou moins satisfaisant pour des raisons esthétiques et perceptuelles. Nous avons vu que l'effet rideau de douche n'est pas ce que l'on peut attendre d'un rendu peinture animé car il gêne la compréhension de l'animation. Cette effet résulte du fait que tous les attributs, hormis la couleur, sont fixés pour la totalité de l'animation. Nous avons remarqué que si l'on relâche ces contraintes en fixant simplement la position des coups de pinceau (pour toute l'animation) en espace image, l'effet rideau de douche est beaucoup moins perceptible dans les mouvements de rotation/zoom mais qu'il subsiste lors des translations. Il serait intéressant d'étudier plus avant les décisions prises par des artistes pour représenter une scène animée en peinture et comprendre les liens avec les mécanismes de perception.

8. Conclusion

Nous avons présenté une nouvelle approche pour le rendu de peinture animée qui permet à l'utilisateur de choisir parmi de nombreux styles tout en conservant une représentation correcte de la scène qui respecte les informations de couleur et de profondeur. Nous nous sommes basés sur les méthodes précédentes pour déterminer le mouvement des coups de pinceau que nous étendons avec des comportements adaptés pour le rendu peinture. Nous avons introduit un algorithme de dessin respectant les contraintes du rendu peinture et de représentation de la scène.

La majorité des opérations se fait sur le GPU ce qui permet d'obtenir des performances interactives. De cette manière, l'utilisateur peut définir son style avec un retour visuel immédiat sur les changements de paramètres.

Références

[BWL04] BAXTER W. V., WENDT J., LIN M. C. : IMPaSTo : A realistic model for paint. In *NPAR'2004 : international symposium on non-photorealistic animation and rendering* (June 2004), pp. 45–56.

[CL06] CHI M.-T., LEE T.-Y. : Stylized and abstract painterly rendering system using a multiscale segmented sphere hierarchy. *IEEE Transactions on Visualization and Computer Graphics. Vol. 12*, Num. 1 (février 2006), 61–72.

[CRH05] COLLOMOSSE J. P., ROWNTREE D., HALL P. M. : Stroke surfaces : Temporally coherent artistic animations from video. *IEEE Transactions on Visualization and Computer Graphics. Vol. 11*, Num. 5 (septembre 2005), 540–549.

[CT05] CHU N. S.-H., TAI C.-L. : Moxi : Real-time ink dispersion in absorbent paper. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2005). Vol. 24*, Num. 3 (août 2005), 504–511.

[Dan99] DANIELS E. : Deep Canvas in Disney's Tarzan. In *SIGGRAPH 99 Conference Abstracts and Applications* (1999), p. 200.

[DS02] DECARLO D., SANTELLA A. : Stylization and abstraction of photographs. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2002). Vol. 21*, Num. 3 (2002), 769–776.

[Hae90] HAEBERLI P. : Paint by numbers : abstract image representations. *Computer Graphics (Proceedings of SIGGRAPH '90). Vol. 24*, Num. 4 (1990), 207–214.

[HE04] HAYS J., ESSA I. : Image and video based painterly animation. In *NPAR'2004 : international symposium on non-photorealistic animation and rendering* (2004), pp. 113–120.

[Her98] HERTZMANN A. : Painterly rendering with curved brush strokes of multiple sizes. In *SIGGRAPH'98* (1998), pp. 453–460.

[Her01] HERTZMANN A. : Paint by relaxation. In *CGI '01 : Computer Graphics International 2001* (2001), IEEE Computer Society, pp. 47–54.

[HP00] HERTZMANN A., PERLIN K. : Painterly rendering for video and interaction. In *NPAR'2000 : international symposium on non-photorealistic animation and rendering* (2000), pp. 7–12.

[HS04] HALLER M., SPERL D. : Real-time painterly rendering for MR applications. In *GRAPHITE '04* (2004), pp. 30–38.

[KLK*00] KLEIN A. W., LI W., KAZHDAN M. M., CORRÊA W. T., FINKELSTEIN A., FUNKHOUSER T. A. : Non-photorealistic virtual environments. In *SIGGRAPH'2000* (2000), pp. 527–534.

[KMM*02] KALNINS R. D., MARKOSIAN L., MEIER B. J., KOWALSKI M. A., LEE J. C., DAVIDSON P. L., WEBB M., HUGHES J. F., FINKELSTEIN A. : WYSIWYG NPR : Drawing Strokes Directly on 3D Models. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2002). Vol. 21*, Num. 3 (juillet 2002), 755–762.

[KWH06] KOLLIOPOULOS A., WANG J. M., HERTZMANN A. : Segmentation-based 3d artistic rendering. In *Rendering Techniques 2006 (Proceedings of Eurographics Symposium on Rendering)* (Nicosia, Cyprus, June 2006), pp. 361–370.

[Lit97] LITWINOWICZ P. : Processing images and video for an impressionist effect. In *SIGGRAPH'97* (1997), pp. 407–414.

[Mei96] MEIER B. J. : Painterly rendering for animation. In *SIGGRAPH'96* (1996), pp. 477–484.

[Pal99] PALMER S. E. : *Vision Science : Photons to Phenomenology*. The MIT Press, Cambridge, Massachusetts., 1999.

[RL00] RUSINKIEWICZ S., LEVOY M. : Qsplat : a multiresolution point rendering system for large meshes. In *SIGGRAPH'2000* (2000), pp. 343–352.

[SD02] SANTELLA A., DECARLO D. : Abstracted Painterly Renderings Using Eye-Tracking Data. In *NPAR'2002 : international symposium on non-photorealistic animation and rendering* (2002), pp. 75–82.

[ST90] SAITO T., TAKAHASHI T. : Comprehensible rendering of 3-d shapes. *Computer Graphics (Proceedings of SIGGRAPH '90). Vol. 24*, Num. 4 (1990), 197–206.

[WXSC04] WANG J., XU Y., SHUM H.-Y., COHEN M. F. : Video tooning. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2004). Vol. 23*, Num. 3 (2004), 574–583.