AFIG 2008 / Toulouse
IRIT Presse

Création de storyboards dynamiques pour la visualisation d'animations

Thierry Stein et Nicolas Holzschuch

Université de Grenoble, CNRS, LJK, INRIA Grenoble RA

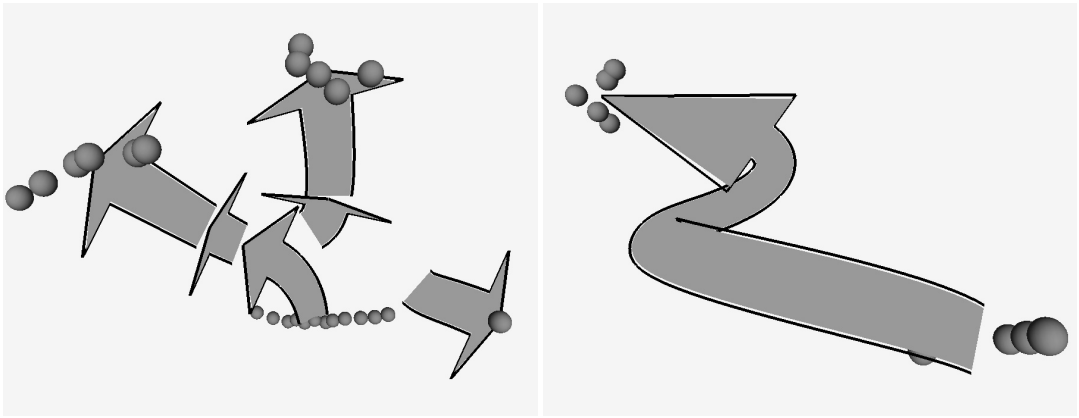

Figure 1: *Exemples d'images obtenues avec notre méthode*

Abstract

Représenter un ensemble complexe de mouvements sous forme condensée, par exemple dans une image, est un problème qui se pose dans de nombreux domaines, allant de la visualisation scientifique à la conception de storyboards ou de bandes dessinées. Une image (espace à deux dimensions) ne peut représenter le mouvement de particules dans l'espace (données 4D) sans perte d'information. Pour compenser cette perte, plusieurs techniques ont été développées, allant de l'ajout d'indices visuels dans une image au découpage du mouvement en une séquence de plusieurs images.

Dans cet article, nous présentons un pipeline pour générer, à partir de données correspondant à un ensemble de mouvements dans l'espace et sur une certaine durée temporelle, un storyboard résumant de manière compréhensible et efficace l'ensemble de l'animation. Notre méthode consiste à grouper les données ayant un mouvement similaire, puis à segmenter ces groupes pour isoler des positions clefs. Enfin, nous effectuons un rendu stylisé de la trajectoire correspondant à chaque segment. L'objectif de notre travail est de permettre une exploration dynamique du storyboard obtenu, de telle sorte qu'un utilisateur puisse observer les données à plusieurs échelles, aussi bien spatiales que temporelles.

Keywords: Informatique Graphique, Visualisation scientifique, Rendu Non-Photoréaliste

1. Introduction

Le 17 Février 2008, à la mi-temps de la rencontre OM-PSG, les télé-spectateurs de Canal+ découvraient le nouveau "gadget" du journaliste Philippe Doucet. La *Palette Plus* (voir figure 2) propose tout un ensemble d'outils pour enrichir une action de jeu à l'aide d'indices visuels. Cette nouvelle interface, déja largement utilisée aux Etats-Unis, permet de mettre en relief de manière concise tout un ensemble de concepts stratégiques et positionnels, offrant ainsi aux spectateurs une nouvelle manière d'observer une retransmission sportive.

L'émergence de tels outils est loin d'être spécifique à la chaîne cablée Française ni même au sport en général. La force d'une image est qu'elle permet de communiquer une information plus rapidement qu'une vidéo et de manière moins ambiguë qu'un texte. A contrario, sa faiblesse provient de la difficulté à représenter un évènement temporel

Figure 2: *Un exemple d'image créée avec la Palette Plus*

au sein d'un support par essence fixe. En effet, une telle opération correspond mathématiquement à une projection de $\mathbb{R}^4$ vers $\mathbb{R}^2$, ce qui induit une double perte d'informations. Compenser cette perte fait encore aujourd'hui l'objet de nombreuses recherches, aussi bien du coté des artistes que des scientifiques.

James E Cutting [Cut02] compare plusieurs techniques pour représenter le mouvement dans une image fixe. Son étude fait ressortir que l'utilisation des lignes d'actions est la méthode qui permet le mieux de créer une image compréhensible en ce qui concerne la précision et la direction du mouvement. Cutting se restreint cependant au cas d'une image fixe, occultant ainsi une partie de ce qui forme la grammaire de la Bande-dessinée. Scott McCloud [Mcc94] démontre que représenter le mouvement **entre les cases** forme l'essence même du neuvième art. Confronté à deux images successives, le cerveau humain cherche naturellement à les interpoler et reconstruit ainsi l'action.

S'inspirant de ces analyses, le travail présenté dans cet article s'attache à utiliser les deux techniques de représentation du mouvement afin d'obtenir un rendu compréhensible d'un ensemble de données animées. Sur le modèle de l'algorithme de Bezerra *et al.* [BEDT08] pour les scènes 3D, nous commençons par partitionner nos données pour obtenir des groupes d'objets qui suivent une trajectoire similaire. Nous segmentons ces données pour éviter les recouvrements puis nous affichons les trajectoires obtenues en utilisant des flèches de mouvement. Nous obtenons ainsi un *storyboard* qui résume de manière efficace une animation complexe. La section 2 passe en revue les différents travaux effectués en synthèse d'images pour remplir cet objectif. Les sections 3 et 4 présentent notre algorithme. Nous terminons en section 5 par un exposé de nos résultats actuels ainsi qu'une reflexion sur les perspectives de ce travail.

2. Travaux précédants

Dans son *Invitation to Discuss Computer Depiction* [Dur02], Frédo Durand distingue la visualisation scientifique de la synthèse d'images en ce que la visualisation s'attache à représenter métaphoriquement des sujets qui ne sont pas par essence visuels. Suivant cette définition, le problème de la représentation du mouvement dans une image se situe alors à cheval entre les deux disciplines. Le mouvement est un élément visuel mais sa visualisation dans une image relève de la métaphore. Le tour d'horizon qui suit se situe donc largement au croisement de ces deux domaines.

2.1. Visualisation Scientifique

La question de représenter le mouvement dans une image se pose en visualisation scientifique sous l'angle de la visualisation de champs de vecteurs. De nombreuses techniques ont été mises au point pour accomplir ce but : advection de textures, extraction de points d'intérêts, calcul de primitives géométriques. L'un des outils les plus courants est l'affichage de *streamlines*, qui peuvent être vues comme des particules lancées dans le champ de vecteurs et dont on visualise la trajectoire. Le problème qui se pose alors est de sélectionner de manière pertinente les *streamlines* à afficher : trop peu fait manquer des mouvements pertinents, un trop grand nombre fournit une *soupe de spaghettis* difficile à interpréter. Les stratégies pour optimiser ce placement sont encore aujoud'hui massivement 2D (Mebarki *et al.* [MAD05]), bien que quelques résultats existent pour la 3D [YKP05]. S'inspirant des techniques de rendu non photoréaliste, Annen *et al.* [ATR*08] ont essayé de définir la silhouette d'un champ de vecteur. Cependant, de même que la silhouette d'un objet ne permet pas à elle seule de percevoir sa forme, les lignes obtenues par leur approche ne rendent pas encore compte de l'ensemble des informations contenues dans le champ de vecteur.

2.2. Informatique graphique

A notre connaissance, le premier article en synthèse d'images qui s'est intéressé à la représentation du mouvement dans une image est celui de Masuch *et al.* [MSS99]. Les auteurs présentent une méthode qui, partant de plusieurs positions clefs, calcule et affiche des lignes d'actions venant enrichir l'image d'un sujet en mouvement. Les travaux de Nienhaus et Döllner [ND05] sont allés plus loin en travaillant à partir du graphe de scène et en créant un graphe d'évènements, offrant une grande souplesse pour la génération de l'illustration finale. Une telle méthode suppose que l'on a accès à une information de structure sur la scène, ce qui n'est pas nécessairement le cas dans toutes les applications.

Plusieurs travaux se sont concentrés sur le cas spécifique de données issues de capture de mouvement. Ainsi, Assa *et al.* [ACCO05] mettent en place une stratégie pour selectionner dans l'animation les positions clefs les plus pertinentes. Leur algorithme se fonde sur une réduction dimensionnelle de l'espace des positions/orientations/vitesses de toutes les

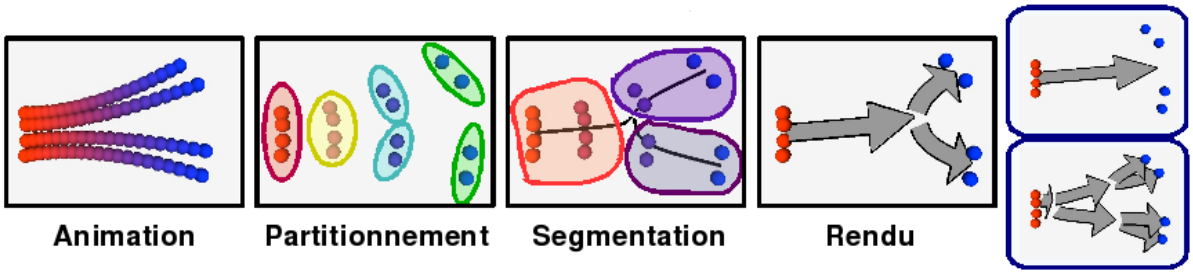

Figure 3: *Aperçu de notre algorithme : nous prenons en entrée un ensemble de particules animées. Les trajectoires sont découpées par l'algorithme de Mean Shift (section 3.2), segmentées (section 4.1), avant d'être affichées sous forme de flèches représentatives du mouvement (section 3.3). L'utilisateur peut faire varier l'échelle de la visualisation, obtenant ainsi soit une indication globale du mouvement (en haut à droite) soit un rendu plus détaillé (bas droite).*

articulations. Ayant projeté ces données dans un espace réduit (typiquement entre 5 et 9 dimensions), les auteurs en extraient un ensemble d'extrema locaux décrivant les positions extrèmes du mouvement. La juxtaposition des poses sélectionnées rend alors compte fidèlement de l'animation originale. Lu et Shen [LS08] ont repris cette stratégie dans le cadre de données volumiques. Associée à un slider temporel, leur technique permet une visualisation interactive de la masse de données initiales.

Bouvier-Zappa *et al.* [BZOP07] proposent une approche complémentaire pour résoudre le même problème. Les auteurs présentent une utilisation combinée de flèches de mouvement, onde de bruit et suivi stroboscopique pour résumer une animation par une image fixe. Le système, partiellement manuel, permet à un utilisateur de sélectionner une pose particulière puis d'afficher un ensemble d'indices visuels dont le rendu s'adapte en fonction du point de vue. La méthode s'appuie de manière extensive sur la hiérarchie du squelette, ce qui rend sa généralisation non-triviale.

Les mêmes questions se posent pour la visualisation de vidéos. Dony *et al.* [DMR05] présentent l'idée du storyboarding inverse et proposent un pipeline pour générer une image qui résume une séquence vidéo. Goldman *et al.* [GCSS06] reprennent cette idée en se basant sur une étude des différentes techniques de storyboarding. Ils proposent de plus d'utiliser les flèches de mouvement calculées comme interface pour sélectionner un instant particulier de la séquence. Ces algorithmes sont limités par la perte d'information sur la profondeur liée à l'utilisation d'une vidéo comme donnée d'entrée. La Palette Plus repose sur une idée semblable. L'information d'entrée est cependant plus riche car le système s'appuie sur une reconstruction 3D effectuée grace à une dizaine de caméras placées autour du terrain. De manière similaire, le système LucentVision [POJC01] permet de visualiser la trajectoire des joueurs et des balles lors d'une rencontre de tennis, fournissant un outil d'analyse puissant pour arbitres, spectateurs, journalistes, joueurs et entraineurs.

3. Notre approche

L'algorithme que nous proposons fonctionne de la manière suivante. L'animation fournie en entrée est partitionnée en groupes de particules ayant un position et une vitesse similaires. Ces données sont ensuite segmentées dans le temps puis les trajectoires rendues sous forme de flèches stylisées. Le storyboard obtenu peut être visualisé à plusieurs échelles en fonction de la résolution choisie par l'utilisateur. La figure 3 résume ce processus. Les étapes de segmentation et de visualisation dynamique n'étant pas encore implémentées, elles seront présentées en section 4.

3.1. Les données d'entrée

La notion de données animées au sens large peut couvrir un grand nombre de représentations et de cas différents. Dans un souci de généralité, nous avons choisi de traiter un ensemble fini et discret de particules, dont on connaît à chaque instant la position, l'orientation et la vitesse.

Cette définition nous permet de considerer de la même manière une simulation de foule, une animation par squelette (chaque particule correspondant à une articulation), ou encore une simulation numérique de type Lagrangienne. Il est aussi possible de traiter des simulations Eulériennes en plongeant au sein de la grille un certain nombre de particules et en suivant leur évolution. Dans ce cas précis, notre méthode peut alors se voir comme une surcouche des techniques à base de *streamlines*.

Formellement, nous considérons un ensemble de n particules $x_i^j \in \mathbb{R}^{d+1}$, x_i^j représentant l'état de la particule i à un instant donné j. L'espace $\mathbb{R}^d$ correspond aux paramètres de position, orientation, vitesse et vitesse angulaire, la dernière dimension étant le temps. D'après Assa *et al.* [ACCO05], cet ensemble de paramètres est un bon descripteur du mouvement global bien que d'autres variables telles que l'accelération puissent être prises en compte. Nous admettrons de plus avoir accès à l'ensemble de l'animation, c'est à dire que

pour un instant t donné, nous avons connaissance des configurations aux instants $t-i$ et $t+i$ et ce quel que soit i.

3.2. Extraction des mouvements globaux

La première étape de notre algorithme consiste à grouper, pour un instant t_i, l'ensemble des particules qui sont à la fois proches dans l'espace et qui suivent globalement le même mouvement. Pour ce faire, nous utilisons un algorithme de *Mean Shift* [CM02] qui permet de grouper des données sans connaissance a priori sur le nombre de groupes à créer. Le principe du *Mean Shift* (voir figure 4) est le suivant : partant d'un ensemble de n points $x_i \in \mathbb{R}^d$ et d'un point source y_0, on construit une série y_j en calculant les moyennes successives des points de données ponderées par un noyau K :

$$y_{j+1} = \frac{\sum_{i=1}^{n} K(y_j - x_i) x_i}{\sum_{i=1}^{n} K(y_j - x_i)} \tag{1}$$

Pour grouper les données x_i, on commmence par placer un point source y_0^i en chaque x_i. Deux points x_{i_1} et x_{i_2} appartiennent alors au même cluster si :

$$\lim_{j \to +\infty} y_j^{i_1} = \lim_{j \to +\infty} y_j^{i_2} \tag{2}$$

Un choix classique pour K est d'employer une gaussiennne :

$$K(t) := \frac{1}{(h\sqrt{2\pi})^d} e^{-\frac{1}{2}\|\frac{t}{h}\|^2} \tag{3}$$

La variance h de la gaussienne définit alors l'échelle à laquelle on effectue le *clustering*, une faible valeur créant de nombreux groupes tandis qu'une forte valeur réunira à partir d'un certain stade l'ensemble des points en un seul *cluster*. En jouant sur cette valeur, on peut faire évoluer le partitionnement en fonction de la distance à l'observateur par exemple. À un observateur lointain correspondra une fonction à rayon large qui aura pour effet de créer de vastes groupes montrant le mouvement général des particules. A contrario, un observateur proche engendrera un rayon plus petit et ainsi des groupes plus restreints, mettant en évidence des mouvements fins qui ne seraient pas visuellement pertinents vus de loin. Cet algorithme permet de n'avoir aucun a priori sur le nombre de groupes. Par aillleurs, contrairement à d'autres techniques de segmentation de type octrees ou KD-trees, le Mean shift permet de créer des clusters de forme arbitraire.

L'avantage du Mean Shift pour notre problème est que cet algorithme ne nous fournit pas uniquement un partitionnement des données mais aussi la moyenne de chacun de ces groupes, c'est à dire que pour chaque groupe et à chaque instant nous disposons des informations de position, orientation et vitesse qui résument au mieux l'ensemble des valeurs de la partition. Un point faible actuel est que cette information est ponctuelle, *i.e.* définie de manière indépendante pour chaque pas de temps, ce qui aboutit lors du rendu à des discontinuités et à la création de mouvements fantômes (voir figure 5).

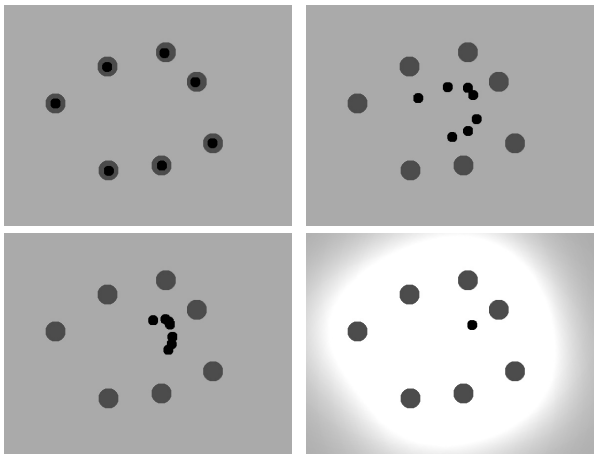

Figure 4: *Les sources (noir) sont positionnées sur chaque point de donnée (gris). Un groupe est créé lorsque les sources convergent vers un même point.*

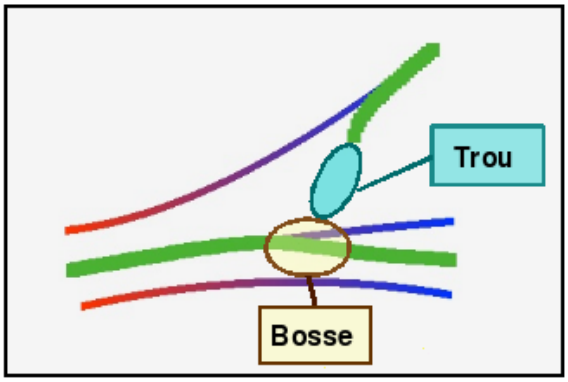

Figure 5: *Deux artefacts causés par notre méthode actuelle : une bosse apparaît lorsque la trajectoire d'une particule commence à diverger de celle des autres ; lorsqu'un deuxième groupe se crée, celui-ci n'a aucun antécédant ce qui produit un trou dans les trajectoires*

3.3. Stylisation

Nos données étant partitionnées, nous disposons à chaque instant de la direction et de la vitesse "moyenne" (au sens du Mean Shift) de chaque groupe. Nous commencons par chaîner ces données pour obtenir une trajectoire continue. Lorsque se produit une rupture du partionnement (séparation ou fusion), nous réinitialisons les trajectoires et en créons de nouvelles pour chaque groupe. Ces trajectoires servent ensuite de support pour l'affichage de flèches de mouvement.

Pour rendre ces flèches, nous calculons deux courbes *Up* et *Down* qui sont des courbes *offset* à la trajectoire. Nous contruisons la pointe de telle sorte qu'elle représente une certaine longueur de la flèche puis nous affichons le tout sur la forme d'une bande de triangles (voir figure 6).

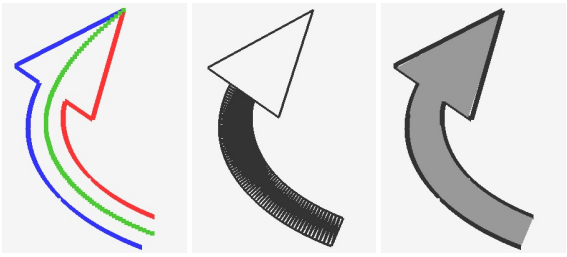

Figure 6: *A partir de la trajectoire (en vert), nous calculons deux courbes Up et Down. La flèche est ensuite affichée dans un style dessiné.*

Pour faire ce calcul nous considérons p_{t-1},p_t et p_{t+1} trois points successifs de la trajectoire et $\vec{V}$ la direction de la caméra. La direction d'*offset* $\vec{b}$ est donnnée par le produit vectoriel entre la tangente de la trajectoire et le point de vue :

$$\vec{b} = (p_t - p_{t-1}) \otimes \vec{V} + (p_{t+1} - p_t) \otimes \vec{V} \qquad (4)$$

Les points des deux courbes *Up* et *Down* sont définis par :

$$\begin{aligned} Up\_t &= p_t + w * \vec{b} \\ Down\_t &= p_t - w * \vec{b} \end{aligned}$$

Ces deux trajectoires définissent la silhouette de la flèche. Le paramètre w correspond à l'épaisseur locale de la flèche, que nous pouvons faire varier en fonction du nombre de particules dans chaque groupe. Tenir compte du point de vue nous permet ainsi de garder la flèche comme faisant face à l'observateur.

Nous calculons les trajectoires *Up* et *Down* jusqu'à un certain instant de la trajectoire (les 5/7^{e}dans notre implémentation). Nous créons alors deux autres points en doublant le coefficient w puis ajoutons le dernier point de la trajectoire initiale pour former la pointe de la flèche de mouvement.

Plusieurs éléments importants rentrent en compte dans l'affichage de ces indices visuels. Notre méthode actuelle ne fait pas ressortir la vitesse du groupe. Ceci pourrait être obtenu soit par un dégradé de couleur, soit en dessinant la flèche non pas sous la forme d'une bande continue mais en pointillés en utilisant un échantillonage temporel régulier (voir un exemple en figure 7). Cette échantillonage ne doit pas être trop important sous peine de perdre l'information de trajectoire.

4. Extensions

Nous décrivons dans cette section la méthode que nous envisageons de mettre en oeuvre pour segmenter temporellement l'animation et pour obtenir une visalisation dynamique. Bien que non-encore implémentés, ces deux modules font partie intégrante de notre pipeline.

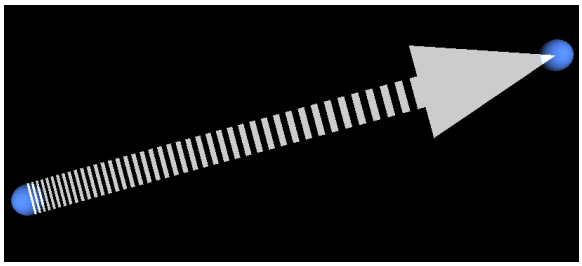

Figure 7: *Exemple d'échantillonage suggèrant la vitesse*

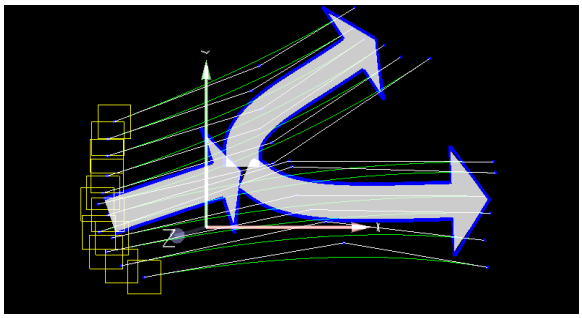

Figure 8: *Un exemple de résultat avec les trajectoires de chaque particule*

4.1. Segmentation temporelle

L'affichage de la trajectoire de chaque cluster donne déja un résultat correct pour certains types de mouvement. Cependant, dans le cas d'animation "stationnaire" ou bien de recouvrement (une personne ou un groupe revenant sur ses pas), la superposition des indices visuels donne un résultat confus. L'objectif de cette étape est donc de segmenter temporellement les groupes afin de lever ces ambiguïtés.

Pour ce faire, nous introduisons une grille 3D régulière représentant l'espace. Chaque voxel de cette grille contient une information de densité qui représente le nombre de groupes ayant occupé ce voxel durant l'animation. Une densité de 1 signifie que la case n'a été occupée que par un seul groupe et à un seul instant, ce qui ne posera pas de difficulté lors de l'affichage final. A contrario, une densité plus élevée signifie soit que (i) plusieurs particules d'un même groupe ont suivi la même trajectoire, (ii) le groupe est revenu sur ses pas, (iii) plusieurs groupes sont passés au même endroit. Si le premier cas ne pose pas de problème visuel, il n'en va pas de même des deux suivants.

Pour segmenter ces données et lever les ambiguïtés liées aux deux cas mentionnés, nous envisageons une approche itérative. Nous considérons l'instant $t_0 = 0$ et nous avançons dans le temps jusqu'à ce qu'une ou plusieurs cellules adjacentes contiennent une information contradictoire, *i.e.* un groupe revient sur ses pas ou bien le mouvement de deux groupes s'intersecte. Nous effectuons alors une coupe temporelle et réinitialisons la grille à ce nouvel instant t_1 et ceci

jusqu'a t_{final}. Cette étape nous fournit en sortie un strip de l'animation de départ, chaque case contenant le mouvement des groupes entre le début de cette image et le début de l'image suivante.

Cette approche naïve risque de créer un trop grand nombre de coupures. Pour obtenir un meilleur compromis, nous envisageons de formuler ce problème sous la forme d'une minimisation d'énergie en essayant de trouver un compromis acceptable entre recouvrements de mouvements et nombre de cases créées.

4.2. Visualisation dynamique

L'informatique graphique permet aujourd'hui de ne plus considerer simplement une image fixe mais d'imaginer des promenades virtuelles à l'intérieur de mondes abstraits. Cette section décrit l'interface que nous souhaitons mettre en oeuvre pour qu'un utilisateur puisse se promener interactivement dans le storyboard de la simulation.

Le résumé se présente sous la forme d'une bande-dessinée de plusieurs cases alignées . L'utilisateur zoome à l'intérieur d'une case et le storyboard se recompose pour mettre en évidence la portion de temps sélectionnée avec plus de détails sur les mouvements décrits, ces détails dépendant de la distance au point de vue. La mise en place d'un tel scénario nécessite deux choses :

1. Une cohérence temporelle lors des modifications du storyboard. La selection d'un intervalle temporel ne doit pas modifier de manière trop violente le partitionnement ni la segmentation mais relever plus du raffinement.
2. Une mise à jour interactive, voire temps-réel, de notre storyboard, ce qui soulève des problèmes aussi bien d'algorithmique que de programmation.

5. Discussion

Nous avons présenté un système pour résumer une animation particulaire sous la forme d'un storyboard visualisable dynamiquement. Notre implémentation actuelle partionne les données en groupes de particules ayant une position et une vitesse proches. Ces partitions nous fournissent des trajectoires que nous affichons sous la forme de flèches de mouvement. Les figures 1 et 8 montrent les résultats que nous obtenons actuellement. Les résultats pour un groupe de particules qui se séparent et pour un mouvement hélicoïdal font ressortir de manière compréhensible le mouvement général de l'animation. La mise en oeuvre des algorithmes présentés en section 4 devrait enrichir ce résulat et permettre de gérer des cas beaucoup plus complexes.

Un paramètre négligé dans cet article concerne la caméra. Le fait de la considerer comme une donnée d'entrée influencerait le partitionnement mais risquerait de nuire à la dynamique du système. Nous pourrions aussi envisager de faire intervenir la caméra en tant que variable supplémentaire, ce qui rajouterait la question du choix d'un point de vue pertinent par rapport aux données animées.

Enfin, nous n'abordons pas la question de la scène dans laquelle se déroule l'animation. Nous traitons pour l'instant un système isolé vivant dans un espace vide. Cependant, des applications réelles (simulation d'incendie par exemple) se déroulent dans un environnement géométrique concret dont il faudrait tenir compte à chaque étape de notre processus pour éviter par exemple de regrouper des particules proches spatialement mais évoluant dans des structures disjointes.

References

[ACCO05] ASSA J., CASPI Y., COHEN-OR D. : Action synopsis : pose selection and illustration. *ACM Trans. Graph. 24*, 3 (2005), 667–676.

[ATR*08] ANNEN T., THEISEL H., RÖSSL C., ZIEGLER G., SEIDEL H.-P. : Vector field contours. In *Graphics Interface 2008* (2008), pp. 97–105.

[BEDT08] BEZERRA H., EISEMANN E., DÉCORET X., THOLLOT J. : 3d dynamic grouping for guided stylization. In *NPAR '08 : 6th International Symposium on Non-photorealistic Animation and Rendering* (2008), ACM, pp. 89–95.

[BZOP07] BOUVIER-ZAPPA S., OSTROMOUKHOV V., POULIN P. : Motion cues for illustration of skeletal motion capture data. In *NPAR '07 : Proceedings of the 5th international symposium on Non-photorealistic animation and rendering* (2007), ACM, pp. 133–140.

[CM02] COMANICIU D., MEER P. : Mean shift : A robust approach toward feature space analysis. *IEEE Trans. Pattern Anal. Mach. Intell. 24*, 5 (2002), 603–619.

[Cut02] CUTTING J. E. : Representing motion in a static image : constraints and parallels in art, science, and popular culture. *Perception advance online publication 31*, 10 (2002), 1165–1193.

[DMR05] DONY R., MATEER J., ROBINSON J. : Techniques for automated reverse storyboarding. *Vision, Image and Signal Processing, IEEE Proceedings - 152*, 4 (Aug. 2005), 425–436.

[Dur02] DURAND F. : An invitation to discuss computer depiction. In *NPAR '02 : 2nd international symposium on Non-photorealistic animation and rendering* (New York, NY, USA, 2002), ACM, pp. 111–124.

[GCSS06] GOLDMAN D. B., CURLESS B., SALESIN D., SEITZ S. M. : Schematic storyboarding for video visualization and editing. *ACM Trans. Graph. 25*, 3 (2006), 862–871.

[LS08] LU A., SHEN H.-W. : Interactive storyboard for overall time-varying data visualization. *Visualization Symposium, 2008. PacificVIS '08. IEEE Pacific* (March 2008), 143–150.

[MAD05] MEBARKI A., ALLIEZ P., DEVILLERS O. : Farthest point seeding for efficient placement of streamlines. *Visualization, 2005. VIS 05. IEEE* (Oct. 2005), 479–486.

[Mcc94] MCCLOUD S. : *Understanding Comics*. Perennial Currents, April 1994.

[MSS99] MASUCH M., SCHLECHTWEG S., SCHULZ R. : Speedlines : depicting motion in motionless pictures. In *SIGGRAPH 99 Conference abstracts and applications* (1999), ACM, p. 277.

[ND05] NIENHAUS M., DÖLLNER J. : Depicting dynamics using principles of visual art and narrations. *IEEE Computer Graphics and Applications 25*, 3 (2005), 40–51.

[POJC01] PINGALI G., OPALACH A., JEAN Y., CARLBOM I. : Visualization of sports using motion trajectories : providing insights into performance, style, and strategy. In *VIS '01 : Proceedings of the conference on Visualization '01* (2001), IEEE Computer Society, pp. 75–82.

[YKP05] YE X., KAO D., PANG A. : Strategy for seeding 3d streamlines. *Visualization, 2005. VIS 05. IEEE* (Oct. 2005), 471–478.