Revue Électronique Francophone d'Informatique Graphique, Volume 7, Numéro 1, pp. 51–65, 2013

Animation réaliste d'une caméra pour la stop motion : de la modélisation à la prise de vue

Laura Saini [1] Gudrun Albrecht [1] Nicolas Lissarrague [2] Lucia Romani [3]

[1]UNIV Lille Nord de France, UVHC, LAMAV-CGAO, FR no.2956, F-59313, Valenciennes, France
[2]UNIV Lille Nord de France, UVHC, CALHISTE, EA 4343, F-59313, Valenciennes, France
[3]Dip. di Matematica e Applicazioni, Università di Milano-Bicocca, Via Cozzi 53, 20125 Milano, Italia

Résumé

L'article présente un nouveau système permettant de produire des mouvements de caméra réalistes pour l'animation stop motion. Le système permettra d'enrichir les logiciels d'animation 3D classiques (comme par exemple Maya *et* 3D Studio Max*) afin de leur faire contrôler des mouvements de caméra pour la stop motion, grâce à l'utilisation d'une interface haptique. Nous décrivons le fonctionnement global du système. La première étape consiste à récupérer et enregistrer les données envoyées par le périphérique haptique de capture de mouvement. Dans la seconde étape, nous traitons ces données par un procédé mathématique, puis les exportons vers un logiciel de 3D pour prévisualiser les mouvements de la caméra. Finalement la séquence est exécutée avec un robot de contrôle de mouvement et un appareil photo. Le système a été évalué par un groupe d'étudiant(e)s du Master "Art plastiques et Création numérique" de l'Université de Valenciennes.*

Abstract

The article presents a new system that allows to create realistic camera movements for a stop motion animation. The system improves traditional 3D software animation programs (for example Maya *and* 3D Studio Max*) for creating stop motion camera movements by using a haptic interface. After describing the whole system, we explain in detail the mathematical processing to obtain different camera movements by using a haptic interface for motion capture. The recorded haptic positions, once elaborated, are exported, frame by frame, to the motion control software, which allows to calibrate the motion control robot, to control the camera settings and, finally, to execute the sequences. A class of students of the "Art plastiques et Création numérique" Master of the University of Valenciennes evaluated the system.*

Mots clé : animation, stop motion, mouvement de caméra, contraintes, simulation réaliste, contrôle de mouvement, paramétrisation.

1. Mouvements de caméra pour la stop motion : état de l'art

Inventée dans les premiers temps de la naissance du cinéma, la stop motion est un type d'animation qui permet de rendre "vivants" presque tous les objets, en donnant l'impression qu'ils bougent par eux-mêmes. L'illusion du mouvement est obtenue en manipulant ces objets dans une succession de poses figées, en photographiant chacune de ces poses et en projetant le résultat comme une séquence continue - le procédé repose sur les mêmes méthodes que le dessin animé, à la différence que les dessins sont remplacés par les photos des objets. C'est un procédé (très) long et (très) fastidieux - un(e) animateur(trice) ne produit que 4 à 12 secondes d'animation par jour [Pur08]. Il est d'autant plus complexe que, si un(e) animateur(trice) fait une erreur sur le plateau, il n'est pas possible de revenir en arrière et de recommencer une animation, celle-ci ne pouvant jamais être reproduite à l'identique.

Avec de telles contraintes, animer une caméra est un exercice périlleux. Il est presque impossible de bouger la caméra image par image le long d'un trajet et d'obtenir un mouvement souple et fluide, principalement parce que la plus petite imprécision produit des secousses très visibles et très peu naturelles. Et, à la différence des tournages classiques, le temps que réclame la stop motion ne permet pas de réaliser plusieurs prises pour n'en retenir que la meilleure. Pendant longtemps, ces contraintes techniques forcèrent les animateurs(trices) à fixer solidement la caméra sur le pla-

teau de tournage et les réalisateurs à se satisfaire de plans fixes. A l'inverse, sur les décors des plateaux de cinéma, depuis W.D. Griffith au début du 20e siècle, l'importance des mouvements de caméra comme partie de l'esthétique spécifiquement cinématographique a été maintes fois soulignée et théorisée. Pour tenter de s'affranchir quelque peu d'une immobilité frustrante, les animateurs(trices) de stop motion essayèrent d'abord de trouver des astuces qui pourraient leur donner davantage de liberté - non sans alourdir la complexité du tournage et augmenter les risques de mauvaises manipulations. Pionnier dans ce domaine, Ray Harryhausen fut ainsi le premier à réaliser un travelling avant dans un court métrage en stop motion, *The Story of the Turtoise and the Hare* (1952).

En 1993, pour échapper à ces difficultés tout en cherchant malgré tout à ce que la caméra puisse être aussi libre de ses mouvements que sur un plateau normal, Henry Selick eut pour la première fois recourt à une caméra contrôlée par ordinateur pour *L'étrange Noël de Mr Jack*. Originellement inventé par John Dykstra pour les effets spéciaux de *La Guerre des étoiles* (Georges Lucas, 1977), le "motion control" ou contrôle des mouvements d'une caméra par ordinateur était jusqu'alors cantonné aux plateaux de tournage «classiques», pour des raisons de coût et d'encombrement principalement. Les dispositifs de contrôle de mouvement tel que le Milo, le Cyclop ou le Juno pésent de 600 Kg à plus d'une tonne, mobilisent plusieurs techniciens et coûtent plusieurs dizaine de milliers d'euros par jour de location. Le budget des productions de film en stop motion étant le plus souvent très limité - bien en-deça de celui des fictions traditionnelles - et se déroulant sur des décors à échelle réduite, il n'est pas étonnant que le recourt à un matériel aussi encombrant et onéreux se fit attendre si longtemps. L'apparition récente de dispositifs de contrôle de mouvement spécifiquement dédiés à la stop motion comme l'Animoko ou la SFH-30 témoignent de l'intérêt que ce domaine de l'animation porte à la possibilité d'élargir ses possibilités de mise en scène. En comparant les deux manières d'animer une caméra en stop motion, il est possible de synthétiser les avantages et les inconvénients de chaque processus d'animation :

- **Animation manuelle** : elle ne permet d'animer la caméra que sur un seul axe de liberté, parfois deux comme le travelling associé à un panoramique dans *The Secret Adventure of Thom Thumb* (David Borthwick, 1993), mais pas davantage sous peine de rendre les manipulations trop complexes ; elle fonctionne bien pour les mouvements à vitesse linéaire mais n'est pas assez précise pour permettre des accélérations, des décélérations ou des mouvements très lents ; la précision humaine ayant ses limites, l'animation finale comporte fatalement des secousses et des tremblements - ce qui par ailleurs confère à la stop motion son cachet. Des films tel que *9,99 $* (Tatiana Rosenthal, 2008), *Fantastic Mr Fox* (Wes Anderson, 2009), *Panique au village* (Stéphane Aubier et Vincent Patar, 2009), des courts métrages comme *Western Spaghetti* (PES, 2008) ou MUTO (BLU, 2008) en donnent de multiples exemples. Dans *Peter and the wolf* (Suzie Templeton, 2006), les animateurs(trices) ont même tenté de simuler une caméra à l'épaule pour une vue subjective du loup.
- **Animation contrôlée par ordinateur** : presque tous les appareils permettant ce type d'animation sont destinés aux plateaux et aux contraintes des prises de vue en temps réel - et sont par conséquent largement surdimensionnés pour la stop motion en termes de coût, de taille, de poids et de vitesse. Quelques exceptions sont apparues depuis peu. Si leur coût et leur taille sont moindre, ils restent cependant encore très élevés et réservés aux budgets conséquents. L'Animoko par exemple représente un coût de 75000€ et pèse 115Kg si bien qu'il n'en existe actuellement que six exemplaires - dont trois appartiennent aux studios Aardman. De plus, dans tous ces systèmes, la trajectoire de la caméra suit des courbes 3D, si bien que la caméra exécute des mouvements parfaits. D'un côté, cet aspect est une bénédiction pour la stop motion puisqu'il empêche toute mauvaise manipulation ou imprécision des animateurs(trices). Mais d'un autre côté, la perfection du mouvement, aussi régulier et désincarné qu'un calcul d'interpolation, s'oppose à l'esthétique de la stop motion. Les films suivants ont eu recours à l'animation de caméra contrôlée par ordinateur : *Wallace and Gromit : A grand day out* (Nick Park, 1989), *Corpse Bride* (Tim Burton and Mike Johnson, 2005), *Wallace and Gromit : The Curse of the Were-Rabbit* (Steve Box and Nick Park, 2005) *Wallace and Gromit : A matter of loaf and death* (Nick Park, 2008), *Coraline* (Henry Selick, 2009), la campagne publicitaire 2010 pour les imprimantes Brother ou très récemment "*Pirates*" (Peter Lord, 2012).

Cette analyse permet de conclure qu'un certain degré et une certaine qualité de secousse est souhaitable car elle apporte du réalisme et participe à l'esthétique de la stop motion. Ce constat a sans doute motivé Peter Lord lorsqu'il déclare : *"Stop-frame is like live music, played on traditional instruments, compared to a studio recording using the finest instruments in the world, all the latest technology and some electronic instruments. The latter is more polished, more perfect, bigger, better, showier - but maybe lacks humanity. Stopframe is much less perfect, much less polished, unrepeatable, inaccurate - in a word, human.", Peter Lord [HD08].*

Cependant, l'animation manuelle image par image est un procédé trop long et trop hasardeux pour permettre d'animer une caméra avec la liberté et la sécurité requises. Notre intérêt se porte donc sur un système de gestion de mouvements de caméra pour la stop motion. Fort du constat fait par les FabLabs (cf. [GN05]) qu'une communauté ayant accès à une nouvelle technologie devient plus créative, le but principal du projet est d'offrir à la communauté de l'animation en volume un environnement de contrôle de mouvement ergonomique et modulaire, capable de simuler un mouvement de caméra comme s'il avait été réalisé par un appareil de prise

de vue réel, et pouvant être (re)produit facilement car ne nécessitant qu'un outillage et des matériaux accessibles, pour un coût aussi modeste que possible.
Au stade actuel de développement du projet, le robot de contrôle de mouvement ne dispose que d'un axe de liberté en translation et de deux axes de liberté en rotation. Pour cette raison, le système d'animation et de contrôle du robot décrit dans l'article se concentre uniquement sur le bruit longitudinal. Il permet de simuler les irrégularités d'accélération et de décélération d'une caméra fixée sur un chariot de travelling. A cette fin, nous analysons d'abord dans la section 2 les outils d'animation 3D existants, puis décrivons le nouveau système dans son ensemble dans la section 3. Dans la section 4 nous détaillons le procédé mathématique permettant d'obtenir des mouvements réalistes (bruités) de caméra à l'aide d'une interface haptique. Dans la section 5 nous présentons une évaluation de notre système par une classe d'étudiant(e)s du master "Art plastique et création numérique" de l'Université de Valenciennes. Finalement, dans la section 6 nous concluons par la présentation des développements futurs de notre système.

2. Outils existants à destination des infographistes pour l'animation de caméra

Cette section présente une description de l'utilisation des principaux logiciels d'animation 3D en remarquant leurs limites pour créer un mouvement de caméra réaliste. Nous ne discutons que des outils utilisés par les animateurs-infographistes pour la réalisation d'images de synthèse et non de ceux qui génèrent des mouvements de caméra temps réel comme dans les jeux ([CAH*96]), ou automatiquement, sans animateur (voir par exemple [HCS96] et [CON08] pour un état de l'art des mouvements de caméra 3D produits sans animateur). En effet, la stop motion, par définition, ne peut être un processus temps réel, et la mise en scène d'un film d'animation requiert de pouvoir prévisualiser par avance le résultat en se conformant à la volonté d'un réalisateur. Un état de l'art de l'animation 3D pour les mouvements de caméra est dressé dans [SALR10] et [SLAR11b], où nous analysons les détails théoriques mathématiques et les solutions existantes. Ces résultats sont ensuite utilisés dans notre système, décrit dans la section 3. La première version de ce système se trouve dans [SLAR11a].

Des logiciels comme *Maya*, *3D Studio Max*, *Lightwave*, *Blender*, *Cinema4D*, *Softimage* et *Houdini* dominent la famille des programmes d'animation 3D. Tous disposent de deux fonctionnalités principales pour l'animation d'objets, à savoir : l'animation par image-clé et l'animation par contrainte de trajectoire.

1. L'**animation par image-clé** repose sur les techniques d'animation classique. Elle est la plus fréquemment utilisée car elle permet de contrôler la caméra soit en modifiant directement sa position dans la scène virtuelle (type "Eyeball in hand", [WO90]), soit en modifiant le cadrage de la scène en visualisant ce que voit la caméra (type "Scene in hand" [WO90]). Son principe est simple : l'utilisateur(trice) choisit les n images importantes, appelées images-clés. Le programme génère ensuite, grâce aux techniques d'interpolation, r images intermédiaires, appelées intervalles. Un objet dans l'espace est représenté par rapport au temps : les positions 3D $\mathbf{p}_i$ de l'objet correspondent au temps t_i pour $i = 1,\ldots,n$. La représentation interne de la trajectoire de l'objet $\mathbf{P}(t)$ est une courbe paramétrée dans l'espace. La figure 1 représente un objet avec $r = 24$ positions dans l'espace. La trajectoire de l'objet dans l'espace est décrite par trois coordonnées $x(t), y(t)$ et $z(t)$. Ces courbes sont visualisées sur le même graphe par le logiciel d'animation (cf. figure 2 où $n = 3$ et $r = 24$). L'animateur ou l'animatrice peut ainsi modifier les coordonnées $x(t)$, $y(t)$ et $z(t)$ de la courbe afin de changer la position de l'objet ou sa vitesse. À ce stade il faut noter que la position et la vitesse ne peuvent pas être modifiées indépendamment l'une de l'autre.
2. L'**animation par contrainte de trajectoire** consiste à construire séparément la trajectoire 3D de l'objet (cf. figure 3) et la courbe de transition temporelle qui contrôle la variation des vitesses de l'objet (cf. figure 4). Par défaut cette courbe est linéaire (cf. 4(a)) et nous ne pouvons pas la modifier, mais nous pouvons ajouter une deuxième courbe qui contrôle et modifie l'accélération et la décélération (cf. figure 4(b)). Dans le cas de l'animation d'une caméra, cette méthode, par son principe même, ne peut

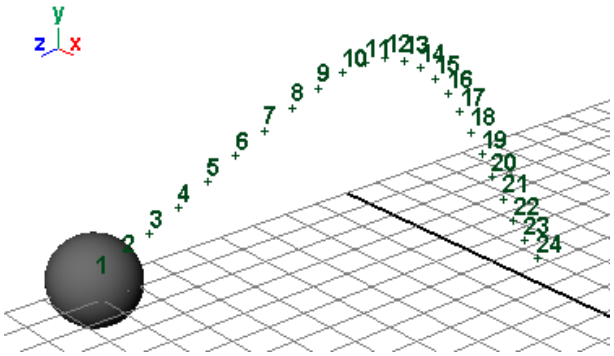


Figure 1: *Représentation de la trajectoire d'un objet dans l'espace 3D dans Maya.*

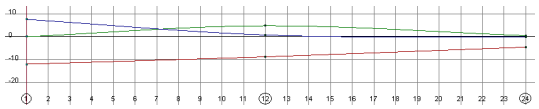


Figure 2: *Éditeur graphique de Maya qui représente la courbe paramétrisée de la figure 1 par ses trois coordonées x, y et z représentées respectivement en rouge, vert et bleu.*

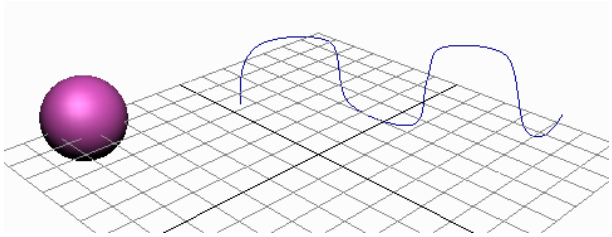
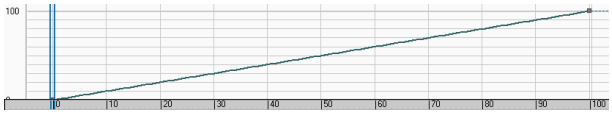
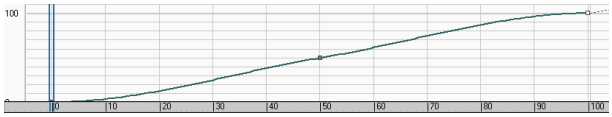


Figure 3: *Visualisation de l'animation par contrainte de trajectoire dans* 3D Studio Max.



(a) Courbe de transition temporelle par défaut et fixe.



(b) Deuxième courbe de transition temporelle modifiable.

Figure 4: *Éditeur de courbes de* 3D Studio Max *illustrant la courbe de transition temporelle associée à la trajectoire de la figure 3.*

servir qu'à modifier la position. L'animation des rotations doit être réalisée avec une autre méthode - le plus souvent par image-clé.

Le tableau 1 présente les avantages et inconvénients de ces deux outils. Il montre qu'aucune des deux méthodes ne permet d'ajouter des contraintes distinctes sur la position et la vitesse d'un objet. En effet, dans le cas de l'animation par images-clé, si l'on ajoute une contrainte sur la courbe de position, la vitesse est elle aussi affectée. Dans le cas de l'animation par contrainte de trajectoire, s'il est aisé d'appliquer une contrainte sur la courbe de vitesse, il n'est en revanche pas possible de l'appliquer sur un seul des axes de la courbe de position. Celle-ci n'est pas accessible sous forme de trois

	Par image-clé	Par contrainte de trajectoire
Séparation position et vitesse	impossible	possible
Contrôle global/local de la trajectoire 3D	contrôle local	contrôle global
Ajout de contraintes	oui	partiellement
Accès aux coordonnées	oui	non

Table 1: *Avantages et inconvénients des deux principaux outils d'animation.*

courbes x, y et z distinctes mais seulement sous forme d'une géométrie 3D (spline). En conséquence, aucune de ces deux méthodes ne permettrait par exemple de contraindre la courbure d'une trajectoire de façon différenciée sur les trois axes et d'ajouter un bruit uniquement sur la vitesse - ce qui est typique des limites et des imperfections d'un appareil de prise de vue réel. Pour surmonter ces inconvénients, nous avons créé un système d'animation basé sur l'animation par image clé, la plus fréquemment utilisée car la plus souple et la plus précise. Notre système permet, de façon sélective, d'ajouter aux courbes d'animation des irrégularités et des contraintes spatiales et temporelles qui simulent le comportement et les limites des appareils de prise de vue cinématographique.

3. Un nouveau système pour les mouvements de caméra en stop motion

Dans cette section, nous décrivons un nouveau système de contrôle de mouvement capable de restituer des contraintes grâce à une interface haptique. Celle-ci permet de produire des imperfections et de simuler le comportement d'accélération et de décélération de certains appareils de prise de vue réel. Nous utilisons une interface de Novint technologies (www.novint.com) : le Novint Falcon (Figure 5), car elle est peu onéreuse et facile à programmer. L'ensemble du système peut être résumé dans les étapes suivantes comme montré dans la Figure 6. Nous commençons par générer une courbe de Bézier indépendamment de toute interaction haptique. Celle-ci représente la trajectoire idéale de la caméra. Une fois les courbes obtenues, on détermine la vitesse avec laquelle la caméra bouge le long de la trajectoire en utilisant l'interface haptique Novint Falcon ("Interface de capture de mouvement"). Celle-ci nous permet, grâce au retour de force, de ressentir l'inertie de l'appareil de prise de vue et ainsi d'intégrer un bruit réaliste représentant les imperfections de la manipulation humaine. D'autre part, la résistance du retour de force nous permet de rester proche de la courbe sans la suivre parfaitement. Ceci permettrait d'intégrer également un bruit transversal correspondant aux imperfections du déplacement spatial des appareils de prise de vue cinématographique. Les contraintes mécaniques de la version actuelle du robot ne permettant pas encore de travailler sur la reproduction du bruit transversal, nous nous concentrons sur le bruit longitudinal. Ainsi, nous effectuons une reconstruction des données enregistrées par l'interface haptique ("Gestion du mouvement de caméra"). Une fois choisie la vitesse, on envoie les points de la paramétrisation à un logiciel qui contrôle le mouvement ("Logiciel de contrôle de mouvement") et qui permet de calibrer le robot et la caméra. La séquence peut être exécutée pour créer la vidéo ("Robot de contrôle de mouvement et caméra numérique"). Pour parvenir à cette fin, l'environnement logiciel est constitué de *Matlab*®, *3D Studio Max* et soit *Dragonframe* soit un logiciel développé en interne. Le système étant encore un prototype, il offre un retour visuel durant certaines phases, mais

pas encore sur la totalité du processus, ce qui est un des objectifs de développement futur.

Figure 5: *Utilisation de l'interface haptique Novint Falcon.*

1. *Interface de capture de mouvement*
   Les interfaces haptiques sont des interfaces de captation spatiale qui peuvent elles-mêmes générer une force sur le point d'entrée (retour de force). Dans notre système, nous pouvons contrôler une courbe paramétrique de Bézier rationnelle cubique (appelée dans la suite courbe de Bézier rationnelle cubique) en utilisant l'interface graphique de [FTA10]. Nous avons choisi ces courbes car elles nous donnent plus de flexibilité, en particulier elles nous permettent de gérer les rotations de la caméra en reproduisant ces mouvements avec un demi-cercle. L'utilisateur(trice) peut être guidé(e) le long de la courbe comme indiqué dans la Figure 7. Son mouvement est ainsi contraint en fonction des limites du robot de contrôle de mouvement de caméra - dans notre cas, un axe de translation et deux axes de rotation. Le point d'interaction x est d'abord déplacé sur le point $p(t_x)$ de la courbe. Comme souvent dans le domaine haptique, une force orientée dans la direction $p(t_x) - x$ est utilisée pour y parvenir. Ensuite, l'utilisateur(trice) peut se déplacer dans le temps le long de la courbe (Figure 8). Voir [SAL*11] pour plus de détails.
   La possibilité de déterminer la position du point d'interaction pour chaque Δt de temps a été ajoutée. Pour respecter la cadence standard à laquelle un périphérique d'affichage diffuse les images, le Δt est réglé à 0.04 secondes soit 25 images par seconde. En appuyant sur le bouton à l'arrière de la poignée, l'enregistrement des mouvements du point d'interaction le long de la courbe est enclenché ; l'utilisateur(trice) peut alors déplacer le point d'interaction le long de la courbe. Lorsque le bouton est relâché, l'enregistrement s'arrête et nous obtenons une séquence de n positions $\mathbf{p}_i^H$, $i = 0, \ldots, n-1$ où H indique la position de l'interface haptique. La séquence $\mathbf{p}_i^H$ représente la paramétrisation de la courbe (Figure 9).

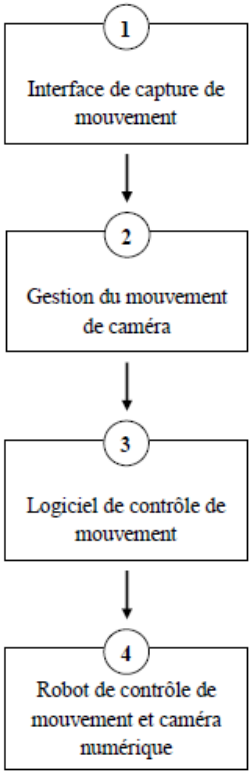


Figure 6: *Schéma du système de contrôle de mouvement.*

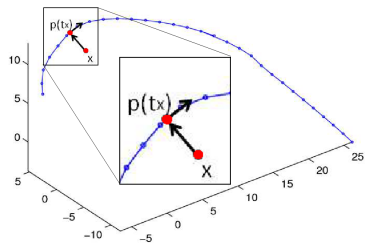


Figure 7: *Point d'entrée haptique x (en rouge) déplacé sur le point $p(t_x)$ de la courbe (en bleu).*

2. *Gestion du mouvement de caméra*
   La séquence $\mathbf{p}_i^H$ de capture de mouvement ne se confond pas exactement avec la courbe cubique paramétrique comme le montre la Figure 10. En outre, la vitesse correspondante est trop bruitée pour obtenir un mouvement de caméra réaliste. Nous voulons que la caméra suive exactement la courbe cubique rationnelle de Bézier $\mathbf{P}(t)$, car le robot est contraint sur la trajectoire donnée. Pour

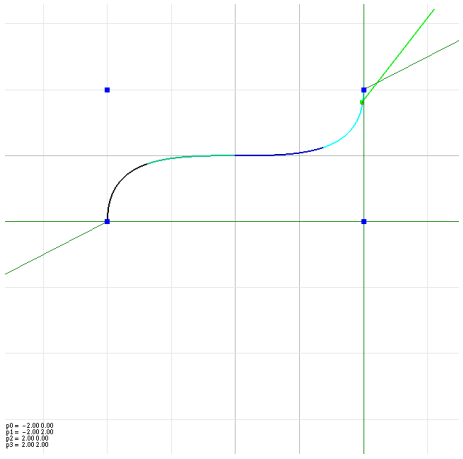


Figure 8: *Interface graphique utilisée par le Novint Falcon qui représente une courbe de Bézier rationnelle cubique.*

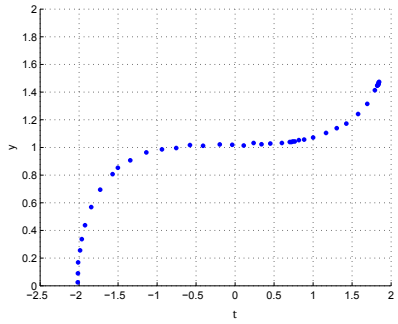


Figure 9: *Visualisation dans Matlab des points* $\boldsymbol{p}_i^H$.

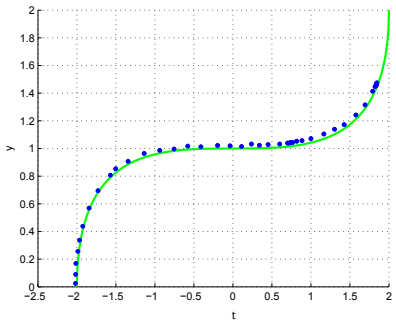


Figure 10: *Une courbe de Bézier rationnelle cubique paramétrisée à l'aide de l'interface haptique. En bleu les positions* $\boldsymbol{p}_i^H$ *obtenues par le Novint Falcon.*

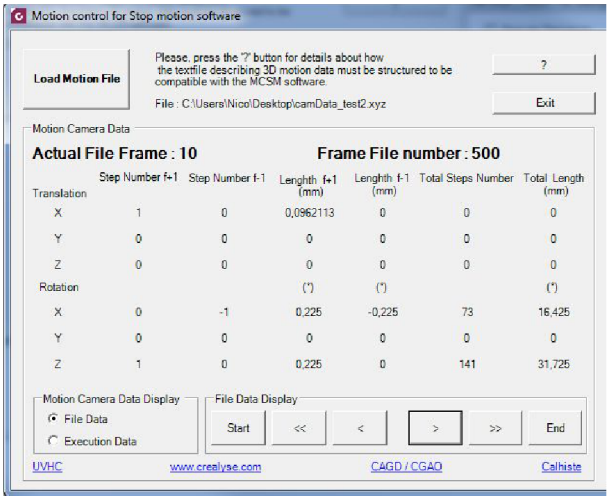


Figure 11: *Interface du logiciel de contrôle du robot et de la caméra. Première partie du panneau de contrôle : elle affiche les informations sur le mouvement de caméra importé depuis le logiciel de 3D (valeurs d'écart de position sur les différents axes, nombre de pas moteur pour les atteindre, etc.). Différents modes permettent d'afficher les informations sans faire bouger le robot, ou au fur et à mesure que celui-ci exécute un mouvement.*

ces raisons, afin d'obtenir une séquence de points sur la courbe et pour simuler différents comportements d'appareils de prise de vue réels, nous devons traiter les positions $\mathbf{p}_i^H$ du point d'interaction. Par calculs mathématiques basés sur la méthode de Newton, décrits dans la section suivante, nous obtenons les nouvelles positions de la caméra virtuelle que nous pouvons exporter image par image au système de contrôle de mouvement.

3. *Logiciel de contrôle de mouvement*
   L'utilisation des données importées permet au logiciel de contrôle de mouvement de calibrer le robot, de contrôler les réglages de l'appareil photo et, finalement, d'exécuter la séquence comme illustré dans les figures 11 et 12.
   Le système est basé sur une interface Arduino (Figure 13), une plateforme de prototypage électronique qui permet, grâce à son microcontroleur, de contrôler le robot avec un signal électronique.
4. *Robot de contrôle de mouvement et caméra numérique*
   Un robot de contrôle de mouvement, sur lequel est fixée une caméra numérique et capable de se déplacer sur un axe de translation et deux axes de rotation, est utilisé pour enregistrer la séquence. Conçu pour être accessible à tous les budgets de production, il peut faire bouger une charge maximale de 2Kg avec une précision de 1/10 de millimètre en translation et 1/10 de degré en rotation (voir la Figure 14 pour une illustration). Le coût du prototype présenté est inférieur à 500€ (hors appareil photo). Il a été assemblé à partir d'éléments tous disponibles au grand public.

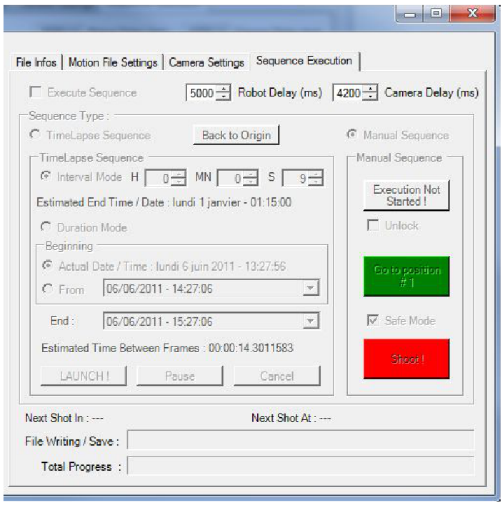


Figure 12: *Interface du logiciel de contrôle du robot et de la caméra. Deuxième partie du panneau de contrôle : elle est divisée en quatre onglet : 1) - File infos : donne les informations relatives à la caméra virtuelle et à la scène 3D importée (focale, cadence d'image, etc.) 2) - Motion file settings : permet le paramétrage des moteurs du robot (attribution à un axe de mouvement, nombre de pas par tour, connexion avec l'Arduino, etc.) 3) - Camera settings : permet de modifier les réglages de l'appareil photo (ouverture, vitesse, ISO, correction d'exposition, etc.) 4) - Séquence exécution : lance l'exécution du mouvement du robot, en synchronisation avec le déclenchement de l'appareil photo. Il est possible de procéder à une exécution automatique (intervalométrie) ou manuelle.*

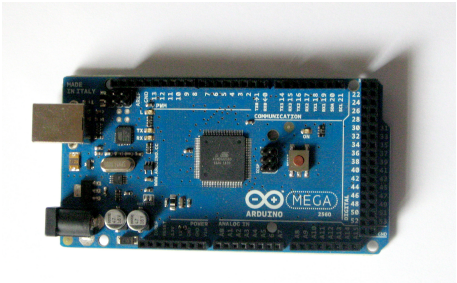


Figure 13: *Arduino Mega 2560.*

5. *Flux de production*
   La façon dont le système présenté est intégré à la création d'une animation commence avec le story board. Celui-ci est la représentation en dessins du découpage plan par plan d'un scénario. Processus central dans la production d'une animation, il est souvent transposé en vidéo afin de pouvoir déterminer finement la durée des plans, la synchronisation et le rythme des mouvements, des dialogues et de la musique. En fonction des indications du story board et des contraintes du dispositif de prise de vue, un mouvement de caméra peut ainsi être créé dans un logiciel de 3D, conforme aux impératifs spatiaux et temporels. Le résultat permet de créer une animatique (story board animé) et de prévisualiser le mouvement, le cadrage, etc. L'animation de la caméra est ensuite exportée vers le logiciel de gestion de l'interface haptique afin de pouvoir y ajouter les imperfections d'accélération et de décélération. Cette animation est enfin récupérée dans le logiciel 3D dans lequel le mouvement peut être synchronisé avec d'autres enregistrements (par exemple translations et rotations). Le mouvement 3D est alors exporté vers un logiciel de stop motion (actuellement *DragonFrame*) qui contrôle le robot et l'appareil de prise de vue : l'animation finale peut être réalisée et le mouvement exécuté image par image.

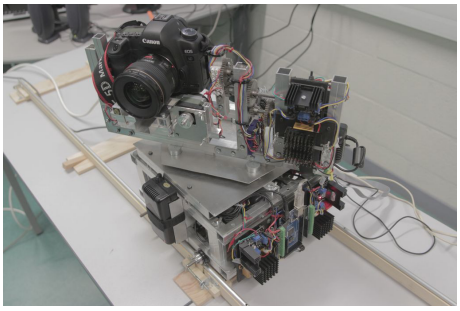


Figure 14: *Robot avec 3 axes de liberté et sa caméra (Canon EOS 5D mark ii).*

La partie correspondant à la gestion du mouvement de caméra nécessite quelques développements, détaillés dans la partie suivante.

4. Procédé mathématique

Cette section présente les détails du procédé mathématique présenté dans la Section 3. En particulier, pour changer la vitesse de notre robot, nous utilisons l'idée d'une courbe de transition temporelle †, qui représente l'abscisse curviligne en fonction du temps. La première étape (*Projection : paramétrisation "haptique" de la courbe*) consiste à projeter la séquence de $\mathbf{p}_i^H$ sur la courbe de Bézier rationnelle cubique. La deuxième étape (*Ajustement : courbe de transition temporelle "idéale"*) détermine une vitesse "idéale" qui maintient la vitesse haptique. Nous pouvons ainsi calculer la paramétrisation idéale de la courbe (*Paramétrisation idéale de la courbe*). Dans la troisième étape (*Mixage : courbe de transition temporelle "intermédiaire"*) nous calculons différentes courbes de transition temporelle entre l'"haptique"

†. Ease Curve

et l'"idéale". Finalement, nous déterminons les paramétrisations correspondantes de la courbe (*Paramétrisation "intermédiaire" de la courbe*). Nous détaillons maintenant le procédé.

1. *Projection : paramétrisation de la courbe "haptique"*
   La séquence de points $\mathbf{p}_i^H(x_i, y_i)$, obtenue à partir de l'utilisation de l'interface haptique, n'est pas exactement sur la courbe de Bézier rationnelle cubique $\mathbf{P}(t)$, où t indique le temps, comme le montre la Figure 15. Pour permettre au robot de suivre la trajectoire initiale, nous devons projeter la séquence respectant la cinématique d'acquisition haptique sur la courbe $\mathbf{P}(t)$, voir Figure 16. Nous connaissons la longueur l de $\mathbf{P}(t)$ où $t \in [0,1]$,

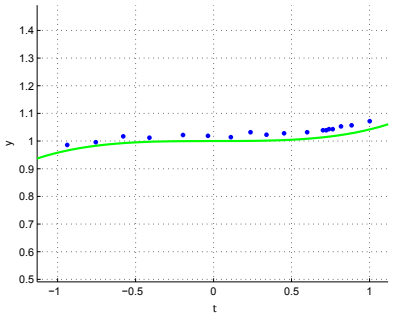


Figure 15: *En bleu : points $\boldsymbol{p}_i^H$; en vert : courbe de Bézier rationnelle cubique $\boldsymbol{P}(t)$.*

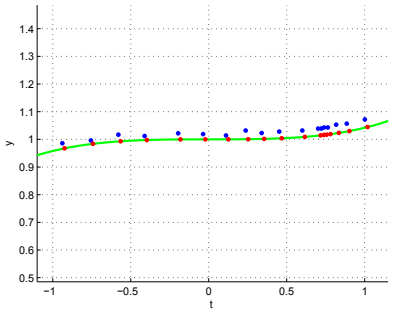


Figure 16: *En rouge : points $\widetilde{\boldsymbol{p}}_i^H$; en bleu : points $\boldsymbol{p}_i^H$; en vert : courbe de Bézier rationnelle cubique $\boldsymbol{P}(t)$.*

comme l'approximation de l'intégrale par une quadrature de Gauss ou une méthode adaptive de Gauss, décrite dans [SALR10]. Ainsi, nous calculons, pour $i = 1,\ldots,n-1$, $sp_i^H = ||\mathbf{p}_i^H - \mathbf{p}_{i-1}^H||$. Nous voulons trouver les valeurs des variables temporelles de la courbe $\widetilde{t}_i \in [0,1]$ avec $i = 0,\ldots,n-1$ qui nous permettent de déterminer les coordonnées des points correspondants sur la courbe $\mathbf{P}(t)$. Un choix pertinent pour la valeur initiale de l'itération est $\widetilde{t}_0 = \frac{||\mathbf{p}_0^H - \mathbf{P}(0)||}{l}$, parce qu'il représente la fraction de l'abscisse curviligne à laquelle le premier point est placé. Nous voulons donc résoudre l'équation suivante :

$$F(\widetilde{t}_i) = \int_{\widetilde{t}_{i-1}}^{\widetilde{t}_i} \left|\left|\frac{d\mathbf{P}}{dt}\right|\right| dt - sp_i^H = 0, \qquad (1)$$

avec $i = 1,\ldots,n-1$. En utilisant la méthode de Newton nous obtenons :

$$\widetilde{t}_i^k = \widetilde{t}_i^{k-1} - \frac{F(\widetilde{t}_i^{k-1})}{F'(\widetilde{t}_i^{k-1})}, \quad k = 1,2,\ldots \qquad (2)$$

où $F(\widetilde{t}_i^{k-1})$ est approximé par des techniques standard et nous pouvons facilement déterminer $F'(\widetilde{t}_i^{k-1})$ grâce à la formule de $\mathbf{P}(t)$ et ensuite calculer $d\mathbf{P}/dt$. Quand F est suffisamment proche de zéro ou le nombre maximal k d'itérations a été calculé, $\widetilde{t}_i^k$ est accepté comme $\widetilde{t}_i$ et l'itération est répétée pour $i = 1,\ldots,n-1$. Si $\widetilde{t}_i > 1$ alors nous imposons $\widetilde{t}_i = 1$. Finalement, la courbe paramétrique $\mathbf{P}(\widetilde{t}_i)$ est évaluée avec les nouvelles valeurs $\widetilde{t}_i$ du paramètre, comme nous pouvons le voir dans la Figure 17. Ce procédé est décrit en détail dans [SALR10].

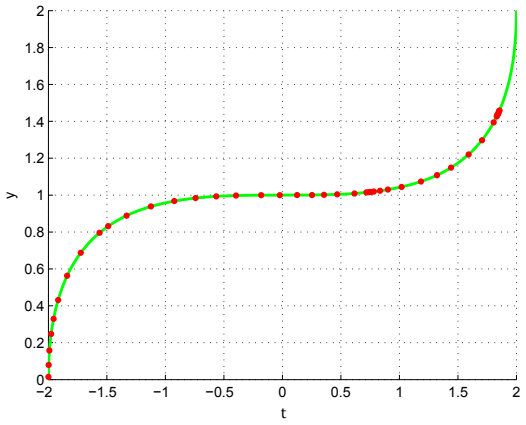


Figure 17: *Projection des positions haptiques $\boldsymbol{p}_i^H$ sur la courbe $\boldsymbol{P}(t)$ résultants dans les point rouges $\widetilde{\boldsymbol{p}}_i^H$.*

2. *Mixage : courbe de transition temporelle "idéale"*
   Une séquence vidéo d'une seconde est composée de 25 images. Nous définissons la variable f comme le nombre d'images par seconde de temps. Dans la Figure 18 nous visualisons la courbe de transition temporelle $S^H(f)$, qui représente l'abscisse curviligne s en fonction de la variable f. Nous calculons

$$s_i^H = \begin{cases} s_{i-1}^H + sp_i^H, & \text{si } x_i > x_{i-1} \\ s_{i-1}^H - sp_i^H, & \text{si } x_i < x_{i-1} \end{cases} \qquad (3)$$

avec $s_0^H = 0$ et x_i l'abscisse de $\mathbf{p}_i^H$ avec $x_i \neq x_{i-1}\ \forall i$. Sur l'axe des ordonnées nous avons les $s_i^H = S^H(f_i)$ et sur l'axe des abscisses les valeurs du paramètre f_i uniformément distribuées avec $f_i \in [0,25]$. En analysant la

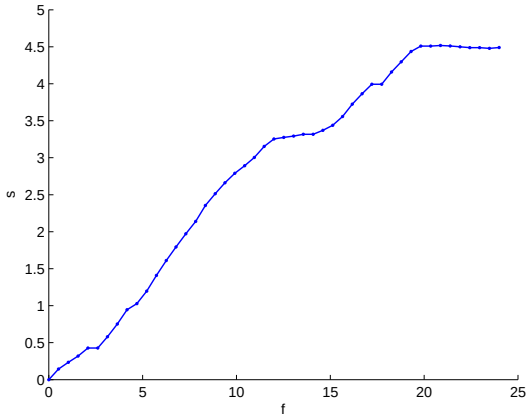


Figure 18: *$S^H(f)$ courbe de transition temporelle de $\boldsymbol{p}_i^H$.*

courbe de transition temporelle nous observons dans la Figure 18 que la vitesse à laquelle la courbe est traversée et, donc le mouvement résultant à partir de l'interface haptique, est trop bruité pour nos objectifs. Ainsi, nous voulons trouver une nouvelle paramétrisation pour notre courbe de Bézier rationnelle cubique qui soit moins bruitée, mais qui respecte encore le mouvement de l'interface haptique. En utilisant la méthode des moindres carrés nous déterminons à partir des données discrètes de la courbe de transition temporelle $S^H(f)$ un polynôme de degré quatre, comme nous l'avons représenté en Figure 19. Ainsi, nous obtenons une vitesse "idéale" $S^I(f_i) = s_i^I$.

3. *Paramétrisation "idéale" de la courbe*
   Comme précédemment, en utilisant la méthode de Newton et en remplaçant sp_i^H par $sp_i^I = |s_i^I - s_{i-1}^I|$ pour $i = 1,\ldots,n-1$, nous obtenons une vitesse "idéale", que nous utilisons pour déterminer une nouvelle séquence de valeurs du paramètre $\widetilde{t}_i^I$. Ainsi, nous pouvons calculer les points correspondants le long de la courbe, pour obtenir la paramétrisation de la courbe "idéale" $\mathbf{P}(\widetilde{t}_i^I)$ (Figure 20).

4. *Mixage : courbe de transition temporelle "intermédiaire"*

   Nous souhaitons disposer de différentes paramétrisations afin qu'un(e) animateur(trice) puisse choisir celle qui s'adapte le mieux à son intention. Nous déterminons une courbe de transition temporelle intermédiaire entre

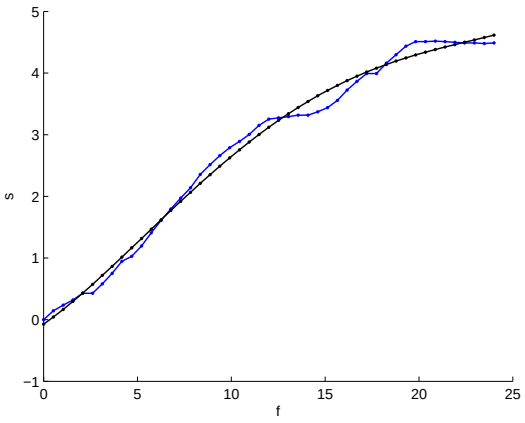


Figure 19: *En noir : courbe de transition temporelle $S^I(f)$ par la méthode des moindres carrés ; en bleu : données discrètes de la courbe de transition temporelle $S^H(f)$ de la Figure 18.*

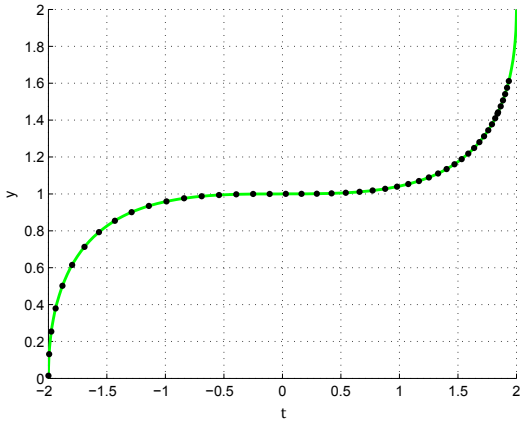


Figure 20: *Paramétrisation "idéale" de la courbe correspondant à la courbe de transition temporelle $S^I(f)$ de la Figure 19.*

l'"idéale" $S^I(f)$ et l'haptique $S^H(f)$. Nous définissons

$$\lambda_i = \alpha_i \cdot \left(\frac{d_i}{||\mathbf{d}||} \right) \tag{4}$$

où $d_i = |s_i^H - s_i^I|$, $\mathbf{d} = (d_0, d_1, d_2, \ldots, d_{n-1})$ et $\alpha_i \in [0, \frac{||\mathbf{d}||}{d_i}]$. Nous calculons une courbe de transition temporelle "intermédiaire" $S^B(f)$ tel que $s_i^B = S^B(f_i)$ par la combinaison barycentrique :

$$s_i^B = (1-\lambda_i)s_i^H + \lambda_i s_i^I, \tag{5}$$

avec $\alpha_i = 1\ \forall i$. Si $\alpha_i = 0\ \forall i$ nous avons la courbe de transition temporelle "haptique" $S^H(f)$ et si $\alpha_i = \frac{||\mathbf{d}||}{d_i}\ \forall i$ nous obtenons l'"idéale" $S^I(f)$. Le choix de λ_i assure que nous respectons la vitesse haptique $S^H(f)$, parce que si s_i^H est proche de s_i^I, l'influence de s_i^I sur s_i^B est négligeable. La courbe $S^I(f)$ change $S^H(f)$ seulement si s_i^H est loin de s_i^I. En Figure 21 nous pouvons voir la courbe de transition temporelle "intermédiaire" $S^B(f)$.

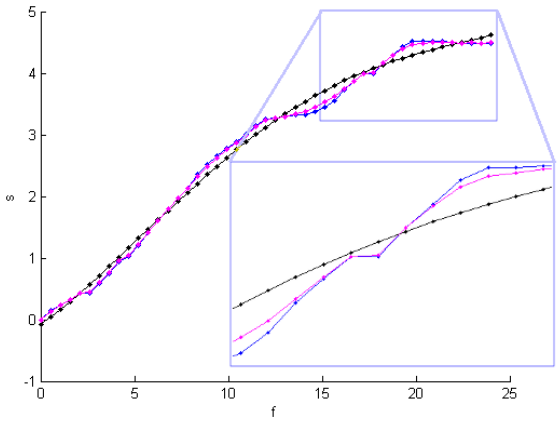


Figure 21: *En magenta : courbe de transition temporelle "intermédiaire" $S^B(f)$; en bleu : courbe de transition temporelle "haptique" $S^H(f)$; en noir : courbe de transition temporelle "ideal" $S^I(f)$.*

5. *Paramétrisations "intermédiaires" de la courbe*
   À nouveau, en utilisant la méthode de Newton avec $sp_i^B = |s_i^B - s_{i-1}^B|$ pour $i = 1,\ldots,n-1$, nous trouvons la nouvelle séquence de paramètres $\widetilde{t}_i^B$. Nous pouvons calculer les points correspondants le long de la courbe pour obtenir la paramétrisation "intermédiaire" de la courbe $\mathbf{P}(\widetilde{t}_i^B)$, voir Figure 22. De plus, l'utilisation du paramètre α_i nous permet de modifier seulement une zone de la vitesse.

Ce procédé, que nous avons implémenté en *Matlab*®, nous permet une grande flexibilité dans le contrôle de la vitesse le long de la courbe, sans modifier la trajectoire. Vu que notre système est capable de bouger sur un axe de translation et deux axes de rotation, nous avons besoin en particulier de contrôler deux courbes : une ligne droite et un demi-cercle. La procédure correspondante est décrite dans les sections suivantes 4.1 et 4.2.

4.1. Translation

Nous voulons contrôler le mouvement linéaire du robot afin de simuler l'inertie et les imperfections d'accélération et de décélération d'un chariot de travelling. Comme nous l'avons représenté en Figure 23, une ligne droite représente

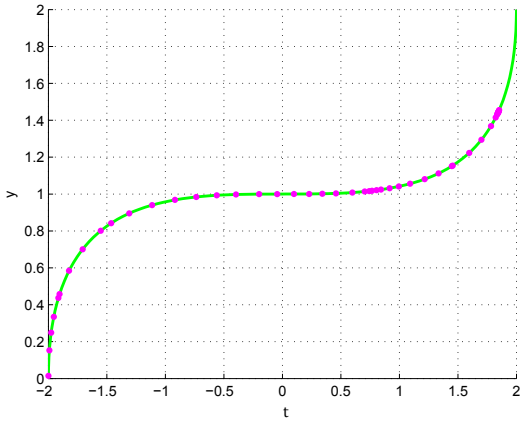


Figure 22: *Paramétrisation "intermédiaire" de la courbe qui correspond à la courbe de transition temporelle $S^B(f)$ en Figure 21.*

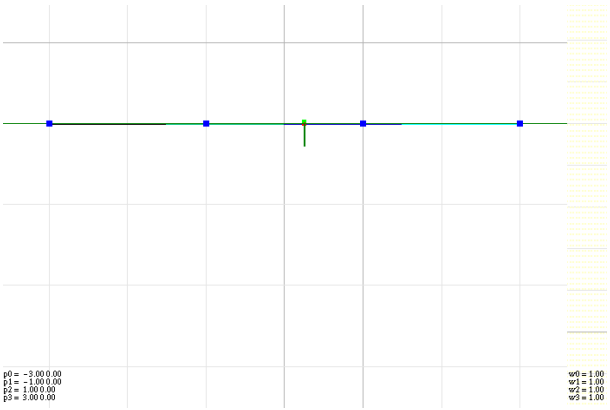


Figure 23: *Trajectoire du robot dans l'interface graphique.*

la trajectoire de notre robot dans l'interface graphique. En particulier, pour avoir une correspondance intuitive avec la trajectoire réelle du robot, nous fixons les points de contrôle de la courbe de Bézier cubique comme :

$$\mathbf{cp}_1 = (-3,0), \quad \mathbf{cp}_2 = (-1,0), \quad \mathbf{cp}_3 = (1,0), \quad \mathbf{cp}_4 = (3,0)$$

et tous les poids égaux à 1. Nous enregistrons les positions de l'interface haptique $\mathbf{p}_i^{H_I}$ concernant la translation. Ensuite, nous suivons la procédure décrite dans la section 4 pour obtenir différents choix de vitesse pour le mouvement de notre robot. Nous importons ces nouvelles paramétrisations de la courbe (haptique, idéale, intermédiaire) dans un script externe de *3D Studio Max*. Nous visualisons la trajectoire et nous simulons le mouvement de la caméra. Un exemple de la paramétrisation de la courbe haptique est vi-

sualisé dans la Figure 24 et ses paramétrisations intermédiaires avec $\alpha_i = \frac{||\mathbf{d}||}{d_i} \cdot 0.9 \; \forall i$ sont montrées en Figure 25.

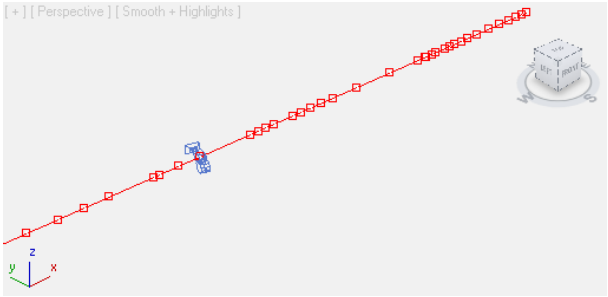


Figure 24: *Projection de la paramétrisation de la trajectoire haptique $\widetilde{\mathbf{p}}_i^{H_t}$ visualisée dans* 3D Studio Max.

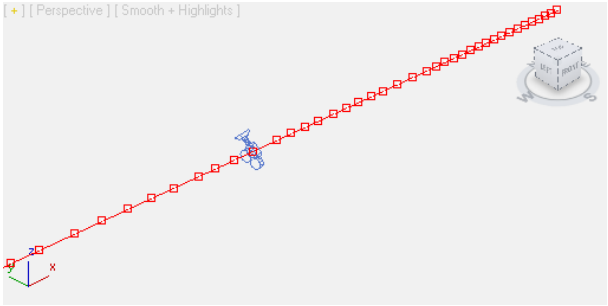


Figure 25: *Paramétrisation "intermédiaire" de la trajectoire $\widetilde{\mathbf{p}}_i^{B_t}$ avec $\alpha_i = \frac{||\mathbf{d}||}{d_i} \cdot 0.9 \; \forall i$ visualisée dans* 3D Studio Max.

4.2. Rotation

La caméra, qui se trouve sur notre robot, peut bouger sur deux axes de rotation. Nous souhaitons contrôler la rotation dans le plan *yz* (Figure 26) et dans le plan *xy* (Figure 27) par l'interface haptique Novint Falcon. Pour les deux axes de rotation, nous imaginons que l'animateur ou l'animatrice manipule les poignées d'une rotule de trépied. Celles-ci sont représentées par l'interface haptique Novint Falcon qu'il ou elle tient dans la main. Pour les deux plans, la caméra peut tourner d'un angle $\omega \in [0, \pi]$. Nous représentons et visualisons cet angle en utilisant un demi-cercle. Dans [PT95] nous trouvons les conditions dont nous avons besoin pour reproduire un cercle par une courbe de Bézier rationnelle cubique. Nous fixons les points de contrôle de la courbe de Bézier :

$$\mathbf{cp}_1 = (1,0), \quad \mathbf{cp}_2 = (1,2), \quad \mathbf{cp}_3 = (-1,2), \quad \mathbf{cp}_4 = (-1,0)$$

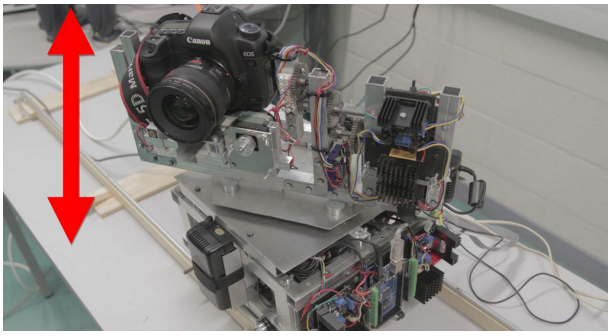


Figure 26: *Visualisation de la rotation de la caméra dans le plan yz.*

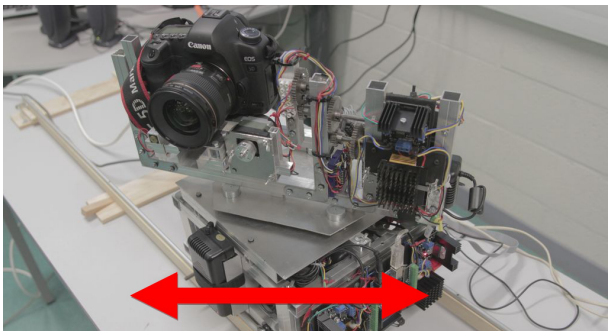


Figure 27: *Visualization de la rotation de la caméra dans le plan xy.*

et les poids comme :

$$w_1 = 1, \quad w_2 = \frac{1}{3}, \quad w_3 = \frac{1}{3}, \quad w_4 = 1,$$

et nous obtenons le demi-cercle en Figure 28. Le demi-cercle représente la rotation dans les deux plans. Le mouvement du Novint Falcon le long du demi-cercle, nous donne l'angle de rotation dans les deux plans de la caméra. La même procédure est appliquée aux deux rotations. Nous déterminons les deux séquences de points $\widetilde{\mathbf{p}}_i^{H_{r1}}$ et $\widetilde{\mathbf{p}}_i^{H_{r2}}$ et nous calculons une nouvelle vitesse. Nous visualisons un exemple de la rotation de la caméra dans le plan *xy* en Figure 29 (paramétrisation "haptique"), en Figure 30 (paramétrisation "idéale") et en Figure 31 (paramétrisation "intermédiaire").

5. Evaluation du système

L'évaluation du système est décisive pour estimer s'il atteint ses objectifs et peut produire des mouvements de caméras réalistes visuellement convaincants, mais également pour identifier les problèmes éventuels et les améliorations possibles. Il est donc essentiel d'obtenir des retours

QUESTIONS \ NIVEAU		1 (-)	2	3	4	5 (+)
Présentation du système (*)		-	-	-	7	7
Capture du mouvement	Translation (**)	-	-	3	8	3
	Rotation (**)	-	-	8	4	2
Utilisation du système (***)		-	-	2	10	2
Questions générales (***)		-	-	-	7	7

(*) CLAIRE
(**)INTUITIVE
(***)EFFICACE

Figure 32: *Résumé du questionnaire rempli par une classe de 14 étudiant(e)s du master "Art plastique et création numérique" du département Arts de l'Université de Valenciennes.*

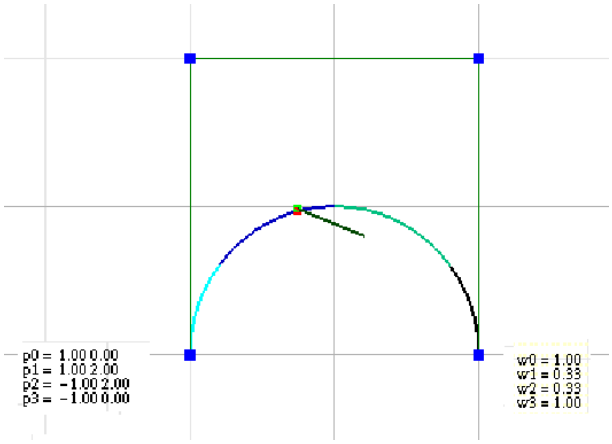


Figure 28: *Demi-cercle représenté par une courbe de Bézier cubique rationnelle et son polygone de contrôle (en vert) dans l'interface graphique du Novint Falcon.*

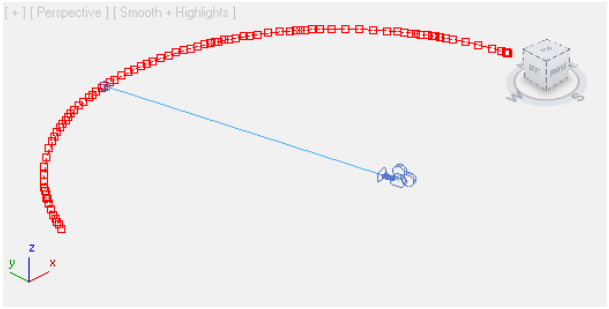


Figure 29: *Paramétrisation haptique de la rotation xy par les points* $\widetilde{\boldsymbol{p}}_i^{H_{r2}}$ *visualisée dans* 3D Studio Max.

des utilisateurs du système. À l'occasion d'un projet avec le département Arts de l'Université de Valenciennes, nous avons demandé à une classe de 14 étudiant(e)s du master "Art plastique et création numérique" de tester l'ensemble du système. Plus précisément, ils ont testé les différentes étapes, depuis la capture du mouvement avec l'interface haptique jusqu'à l'importation des données dans le logiciel de contrôle de mouvement. Ils n'ont pas directement contrôlé le robot mais ont visualisé dans 3Dsmax les différents mouvements : le mouvement brut enregistré par le périphérique haptique et ses dérivations mathématiques. Les mouvements enregistrés sont actuellement utilisés pour la réalisation d'un clip vidéo en stop motion. Les objectifs et le fonctionnement du système ont été présentés préalablement à son utilisation. Puis chaque étudiant(e) a eu la possibilité de tester l'interface haptique pour déterminer la translation et la rotation du robot et visualiser le résultat dans 3Dsmax. Nous avons réuni le plus d'informations possibles sur le système en demandant aux étudiant(e)s de remplir un questionnaire portant sur l'efficacité, la facilité d'utilisation et la précision des mouvements en rotation et translation. Plus précisément, le questionnaire était composé de quatre parties : "Présentation du système", "Capture du mouvement (translation et rotation)", "Utilisation du système" et "Questions générales". La Figure 32 présente un résumé du question-

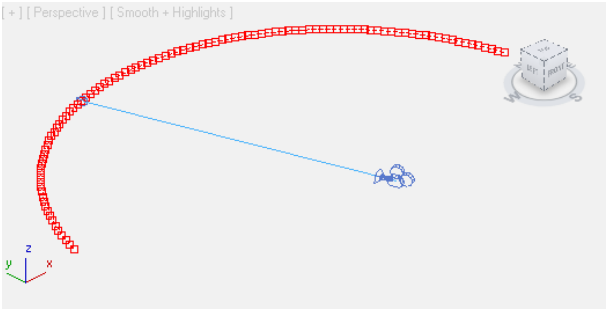


Figure 30: *Paramétrisation idéale de la rotation xy par les points* $\widetilde{\boldsymbol{p}}_i^{I_{r2}}$ *visualisée dans* 3D Studio Max.

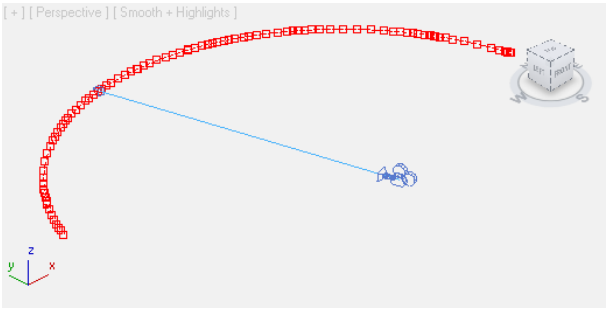


Figure 31: *Paramétrisation intermédiaire de la rotation xy par les points* $\widetilde{\boldsymbol{p}}_i^{B_{r2}}$ *visualisée dans* 3D Studio Max.

naire avec ses questions et ses résultats. Dans la première partie, nous avons demandé si la présentation était claire et s'il y avait besoin de davantage de précisions. Pour tous les étudiant(e)s la présentation était claire (très claire pour 7 d'entre eux et claire pour les autres), bien que les mathématiques ne soient pas leur domaine. Ils ont seulement indiqué qu'il fallait mieux souligner le fait que la gestion du mouvement en translation se faisait séparément de celle en rotation. Dans la seconde partie, nous avons demandé si la capture de mouvement en translation et en rotation était intuitive (très intuitive, intuitive, modérément intuitive, peu intuitive, pas intuitive). Pour tous les étudiant(e)s, la capture de la translation est aisée (très intuitive pour 3 étudiant(e)s, intuitive pour 8 étudiant(e)s et modérément intuitive pour 3 étudiant(e)s). Il a été suggéré d'améliorer l'interface de visualisation 3D en ajoutant soit une icône de caméra soit le tracé de la main afin d'obtenir une meilleure immersion. Inversement, la capture de mouvement des rotations est moins intuitive (modérément intuitive pour 8 étudiant(e)s, intuitive pour 4 étudiant(e)s, très intuitive pour deux étudiant(e)s). Les observations des étudiant(e)s montrent que les mouvements de rotation ne sont pas assez simples pour exécuter un mouvement correctement. Cependant, la rotation sur le plan yz (panoramique haut/bas) est plus facile que celle sur le plan xy (panoramique droite-gauche) grâce à la sensation de tenir dans sa main une rotule de trépied plutôt que la caméra elle-même. En conséquence, le mouvement obtenu est inversé car lorsqu'on bouge la rotule d'un trépied vers le haut, la caméra fixée dessus bouge vers le bas et vice versa. Malgré tout, le traitement mathématique, dans les deux cas, est jugé très approprié (très efficace pour 7 étudiant(e)s, efficace pour les autres). Les étudiant(e)s ont estimé que le mouvement final correspond à celui réalisé avec l'interface haptique. La troisième partie concerne l'évaluation de l'utilisation du système dans sa globalité. Nous avons demandé si elle était adéquate, quelles étaient ses faiblesses et quels aspects pourraient être améliorés. L'ensemble des évaluations indique que le système est efficace (très efficace pour 2 étudiant(e)s, efficace pour 10 étudiant(e)s et modérément efficace pour 2 étudiant(e)s). Il fonctionne correctement et permet de reproduire facilement des mouvements de caméra réalistes. Le résultat n'a pas l'aspect d'un mouvement généré par ordinateur car l'interface haptique permet d'ajouter du bruit de façon naturelle. En revanche, une moitié des étudiant(e)s ont réalisé le mouvement voulu avec quelques difficultés. La raison principale est leur manque d'expérience à manipuler une interface haptique. Selon eux, avec un minimum d'entrainement et de répétition avec ce périphérique, ils seraient capables d'améliorer le résultat. Le principal problème, technique et ergonomique, concerne l'échelle de la représentation de la trajectoire (l'échelle du mouvement réel est de 6 mètres tandis que la capture de mouvement se fait sur 6 centimètres). Pour cette raison, contrôler et gérer la vitesse le long de la courbe est difficile. Les étudiant(e)s ont suggéré d'amplifier à la captation la longueur de la translation et le rayon de rotation. Ils ont enfin souligné qu'il y avait trop d'étapes avant de pouvoir visualiser le mouvement dans 3Dsmax. Une autre question concernait le mouvement après le traitement mathématique. Ils ont trouvé ce processus efficace (très efficace pour 7 étudiant(e)s, efficace pour les autres étudiant(e)s). Ils ont remarqué que le mouvement haptique devait être retravaillé mais que le résultat de ce traitement respectait ce qu'ils voulaient obtenir. En particulier, ils ont estimé que le mouvement le plus correct était celui créé avec $\alpha_i = \frac{||\mathbf{d}||}{d_i} \cdot 0.9 \; \forall i$ qui est un mouvement proche du mouvement idéal comme montré dans la Figure 25. Finalement, nous leur avons demandé de préciser ce qu'ils avaient apprécié ou pas de l'ensemble du système, et s'ils avaient des remarques générales. L'ensemble des étudiant(e)s a trouvé le nouveau système de contrôle de mouvement intuitif et innovant, car il permet de simuler un mouvement de caméra aussi réaliste que ceux réalisés au cinéma et évite les inconvénients des logiciels d'animation 3D. Surtout, il convient à de nombreux types d'utilisateurs car il est simple d'utilisation et abordable.

6. Conclusion

Les éléments détaillés précédemment nous amènent à la conclusion que notre système peut être proposé comme une nouvelle méthode aidant les animateurs(trices) à réaliser des mouvements de caméra réalistes pour l'animation stop motion. Pour les objectifs futurs, nous voudrions que les différentes étapes actuellement réalisées dans plusieurs logiciels puissent être intégrées au sein d'un unique programme, et également améliorer la précision des mouvements de rotation du robot. Le premier objectif futur est souligné par les résultats de l'évaluation : pour être efficace dans un environnement de production, le procédé d'animation de la caméra doit être concentré dans un seul logiciel (création du mouvement, gestion de l'interface haptique et ajout des imperfections, modifications éventuelles du mouvement, exportation des données). Le second objectif majeur est d'étendre les possibilités de mouvements du robot de contrôle à trois axes de liberté en rotation et en translation. Il sera ainsi possible de simuler les différents dispositifs de prise de vue (dolly, Louma, grue, steadycam, etc.) en même temps que leurs contraintes spécifiques (bruit spatial et temporel, limites de débattement, inertie, etc.).

7. Remerciements

Ce travail a été financé par une allocation de recherche de la Région Nord Pas de Calais (France) et de l'Université de Valenciennes et du Hainaut-Cambrésis. Nous sommes reconnaissants de ce support financier. Les auteurs remercient les rapporteurs pour leur lecture attentive de l'article et pour leurs précieux commentaires.

Références

[CAH*96] CHRISTIANSON D., ANDERSON S., HE L., SALESIN D., WELD D., COHEN M. : Declarative camera control for automatic cinematography. *In Proceedings of the National Conference on Artificial Intelligence* (1996), 148–155.

[CON08] CHRISTIE M., OLIVIER P., NORMAND J.-M. : Camera control in computer graphics. *Computer Graphics Forum. Vol. 27*, Num. 8 (2008), 2197–2218.

[FTA10] FÜNFZIG C., THOMIN P., ALBRECHT G. : Haptic manipulation of rational parametric planar cubic using shape constraints. *SAC '10 Proceedings of the 2010 ACM Symposium on Applied Computing,* (2010), 1253–1257.

[GN05] GERSHENFELD NEIL F. : *The coming revolution on your desktop - from personal computers to personal fabrication*. Basic Books, USA, 2005.

[HCS96] HE L.-W., COHEN M. F., SALESIN D. H. : The virtual cinematographer : a paradigm for automatic realttime camera control and directing. *SIGGRAPH '96, In Proceedings of the 23rd annula conference on Computer graphics and interactive techniques. Vol. 8*, Num. 4 (1996), 217–224.

[HD08] HARRYHAUSEN R., DALTON T. : *A Century of model animation*. Aurum Editions, 2008.

[PT95] PIEGL L., TILLER W. : *The NURBS book*. Springer-Verlag, London, UK, 1995.

[Pur08] PURVES B. : *Stop motion : Passion, Process and Performance*. Elsevier, Focal Press, p. 11, (Section 1, Chapter 2 : "Evolving into stop motion - Messing with time"), 2008.

[SAL*11] SAINI L., ALBRECHT G., LISSARRAGUE N., ROMANI L., FÜNFZIG C., BÉCAR J. : Animation 3d : mouvements de caméra réalistes pour la stop motion. *Dans Actes de la 11ème conférence internationale H2PTM (Hypertextes et Hypermédias, Produits, Outils et Méthodes),* (Hypermédiaset pratiques numériques, Université de Metz, Hermes Science/Lavoisier, Octobre, 2011), 137–148.

[SALR10] SAINI L., ALBRECHT G., LISSARRAGUE N., ROMANI L. : Animation 3d : le problème de mouvement de caméra. *Dans Actes des Journées du Groupe de Travail en Modélisation Géométrique (GTMG),* (Dijon, 2010), 69–76.

[SLAR11a] SAINI L., LISSARRAGUE N., ALBRECHT G., ROMANI L. : Stop-motion animation : towards a realistic camera movement control. *Dans Actes de ISEA 2011, The 17th International Symposium on Electronic Art,* (http ://isea2011.sabanciuniv.edu, Sabanci University, Istanbul, Septembre, 2011).

[SLAR11b] SAINI L., LISSARRAGUE N., ALBRECHT G., ROMANI L. : Stop-motion animation : from a state of

the art to an ideal process. *Matapli (édité par la Societé de Mathematiques Appliquées et Industrielles (SMAI), Paris), bulletin 95,* (Juin 2011), 37–52.

[WO90] WARE C., OSBORNE S. : Exploration and virtual camera control in virtual three dimensional environments. *In I3D '90 Proceedings of the 1990 symposium on Interactive 3D graphics, ACM New York* (1990), 175–183.