4. L'ANIMATION PAR ORDINATEUR

4.1 Introduction à l'animation par ordinateur

4.1.1 Les rôles de la simulation graphique et de l'animation par ordinateur

Nous avons vu dans les cours précédents comment construire et modéliser des objets graphiques. Nous avons vu que les objets pouvaient être définis dans un monde à deux dimensions ou dans un monde à trois dimensions. Dans ce dernier cas, nous avons introduit le concept de caméra synthétique qui permet de visualiser la scène tridimensionnelle. Ensuite, nous avons étudié comment recréer des situations d'éclairage par des sources lumineuses avec de l'ombre portée.

Ces techniques sont très importantes puisqu'elles permettent de visualiser n'importe quelle situation géométrique, physique, chimique à **un instant donné**. Mais l'intérêt de beaucoup de phénomènes réside dans leur évolution au cours du temps; par exemple: mouvement de systèmes électromécaniques (robots), réactions chimiques, mouvements de fluides, de gaz (nuages), conduction de la chaleur. L'expérimentation des phénomènes est souvent coûteuse, voire impossible; par exemple, collision de véhicules, explosions. Il est plus facile et plus rentable de procéder à des simulations graphiques des phénomènes.

La **simulation graphique** sur ordinateur repose en fait sur les techniques d'animation.

Une **animation** consiste en la modification d'une scène au cours du temps. Prenons le cas général d'une scène tridimensionnelle; nous pouvons dire qu'elle est composée de 3 types d'entités: les objets, les caméras et les lumières. Chaque entité a des caractéristiques qui peuvent évoluer au cours du temps de façon arbitrairement complexe:

1) **pour les objets**:
- la position (automobile)
- l'orientation (bras de robot)
- la taille (croissance)
- la forme (nuage, coeur humain)
- la couleur (plaque de cuisinière électrique qui chauffe)
- la transparence (simulation de brouillard évoluant)

2) **pour les caméras**:
- la position de l'observateur (simulation de vol)
- le point d'intérêt
- l'angle de vue (zoom in).

3) **pour les sources de lumières**
- l'intensité
- la position (simulation de phares de voiture).

4.1.2 L'animation en temps réel et l'animation image par image

On distingue généralement deux types d'animation: l'animation en temps réel et l'animation image par image. Idéalement, la plus intéressante est l'animation en temps réel où l'ordinateur calcule les mouvements et les transformations suffisamment vite pour que l'usager devant son terminal graphique

puisse voir ces mouvements et ces transformations. C'est ce qu'on observe par exemple dans les jeux vidéos ou dans les simulateurs de vol. Dans le cas des jeux vidéos, l'animation est parfois très simple (2 dimensions, en lignes); donc même un micro-ordinateur est capable de calculer suffisamment vite les transformations. Pourtant, on trouve des animations complexes sur des stations dédiées comme la Playstation de SONY ou le Nintendo 64. Mais, généralement de nombreux mouvements et points de vue de caméras sont précalculés. Pour les simulateurs de vol, on utilise du matériel souvent très coûteux et on simplifie les paysages (bâtiments simples, herbe unie ou même montagnes en 2 dimensions).

L'animation par ordinateur, image par image, correspond à l'animation traditionnelle. Il faut d'abord faire calculer les images, les enregistrer sur bande vidéo (ou film) puis les visualiser ou projeter à une cadence rapide (p.e. 25 images par seconde pour le vidéo PAL/SECAM). Les images calculées peuvent prendre une fraction de seconde à plusieurs heures.

Prenons un exemple: on veut déplacer le long de l'axe X, une automobile placée en <5,0> de 100 mètres en 5 secondes. On suppose que l'on tourne la séquence en 25 images par seconde, ce qui donne 125 images pour les 5 secondes. On pourra donc programmer cette animation de la façon suivante:

en temps réel:

```
PASX:=100 / 125;
créer AUTO;
placer AUTO (<5,0>);
dessiner AUTO;
pour IMAGE:=1 a 125
    attendre;
    effacer AUTO;
    translater  AUTO (<PASX , 0>);
    dessiner AUTO;
```

image par image

```
PASX:=100 / 125;
créer AUTO;
placer AUTO (<5,0>);
dessiner AUTO;
pour IMAGE:=1 a 125
    enregistrer l'image;
    attendre;
    effacer AUTO;
    translater  AUTO (<PASX , 0>);
    dessiner AUTO;
```

Dans les années à venir, on peut espérer obtenir des animations de plus en plus complexes dans des temps de plus en plus courts, grâce à la recherche dans le domaine du parallélisme et des multiprocesseurs. En effet, comme on l'a déjà vu dans le cours, certains algorithmes de synthèse d'images comme le lancer de rayons et les algorithmes de balayage se prêtent bien à un découpage

en tâches parallèles. De plus, l'animation comme on va le voir dans la suite peut être considérée comme un ensemble d'activités parallèles avec ou sans communications entre elles.

4.1.3 Histoire des systèmes d'animation

Les premiers systèmes d'animation étaient des systèmes basés sur des commandes et non-graphiques. Mais, très tôt apparaissent des systèmes en temps réel permettant de définir interactivement des trajectoires et le mouvement. On voit aussi paraître les palettes électroniques, des facilités pour définir des mouvements avec une tablette graphique et les systèmes commerciaux d'animation 2D avec possibilités temps réel et interaction avec la souris (Macromedia DIRECTOR)

Les premiers films 3D par ordinateur sont produits avec de la programmation, tandis que les premiers système d'animation 3D interactifs (BBOP, TWIXT) sont basés sur des positions-clés. Ces techniques deviennent des outils standards dans les produits commerciaux comme Alias, Wavefront, Explore ou SoftImage.

Mais, les systèmes à position-clés sont limités pour des mouvements complexes d'où des systèmes à base de commandes sont développés pour permettre la définition de chorégraphies impliquant caméras, acteurs et lumières. Puis, c'est l'introduction de systèmes basés sur la physique; généralement ils sont à menus et sans interaction graphique.

4.2 Les principales méthodes d'animation par ordinateur

Une séquence d'animation par ordinateur est obtenue par une série d'images produites par ordinateur selon les directives de l'animateur. Nous pouvons distinguer plusieurs méthodologies générales pour la création de ces séquences:

- animation basée sur la capture de mouvements
- interpolation des formes ou animation par images-clés
- animation par interpolation paramétrique
- animation procédurale ou algorithmique.

4.2.1 Animation basée sur la capture de mouvement

Elle est basée sur les mesures et l'enregistrement des actions directes d'un acteur pour l'analyse et la réplication immédiate ou retardée. Ceci implique de faire correspondre le mouvement du personnage digital aux mesures de mouvement. La correspondance peut être **directe**: p.e. bras humain contrôlant le mouvement du bras de l'acteur de synthèse ou **indirecte**: p.e. mouvement de souris contrôlant les yeux d'un personnage et la direction de sa tête.

Il y a 3 sortes de systèmes: mécanique, magnétique et optique.

4.2.1.1 Systèmes mécaniques ou marionnettes digitales

Les marionnettes digitales permettent l'animation de personnages 3D par l'utilisation d'un certain nombre de dispositifs d'entrée temps-réel: souris, joysticks, datagloves, clavier, boites à boutons. L'information fournie par la manipulation de tels dispositifs est utilisée pour contrôler la variation de paramètres au cours du temps pour chaque caractéristique de l'acteur. Typiquement, les paramètres sont des angles d'articulation pour l'animation d'acteurs.

4.2.1.2 Systèmes optiques de capture de mouvements

Ils sont basés sur de petits capteurs réflectifs appelés **marqueurs** et attachés au corps de la vraie personne et focalisés sur la scène. En repérant les positions des marqueurs, on peut obtenir les positions correspondantes pour le modèle animé.

p.e. on attache de petits capteurs aux articulations d'une personne et on enregistre la position de ces capteurs selon plusieurs directions.

d'où la reconstruction des positions 3D de chaque point-clé à chaque instant. L'avantage est la liberté de mouvement, cela ne demande aucun câblage.

Il y a un problème quand il y a occlusion c'est-à-dire un manque de données à cause de marqueurs cachés, p.e. quand la personne est couchée sur le dos. Un autre problème est qu'il est difficile de distinguer 2 marqueurs quand ils sont trop proches lors d'un mouvement.

Les problèmes peuvent être minimisés en ajoutant plus de caméras, mais le coût devient prohibitif. La plupart des systèmes opèrent avec 4-6 caméras.

Exemple de systèmes optiques: Elite, MultiTrax.

4.2.1.3 Systèmes magnétiques de capture de mouvements

Ils demandent qu'une personne porte un ensemble de capteurs magnétiques. Ces capteurs sont capables de mesurer leur relation spatiale par rapport à un transmetteur magnétique centralisé. La position et l'orientation de chaque capteur sont utilisées ensuite pour faire mouvoir l'acteur digital. Il y a un besoin de synchronisation des récepteurs. Les données en provenance des récepteurs et transmises à l'ordinateur sont les positions et orientations de chaque capteur.

Pour le mouvement du corps humain, 11 capteurs sont nécessaires:

- un sur la tête
- un sur chaque bras
- un sur chaque main
- un autre au centre de la poitrine
- un sur le bas du dos
- un à chaque cheville
- un sur chaque pied

On utilise la cinématique inverse pour calculer le reste de l'information nécessaire.

Les systèmes les plus populaires sont: Polhemus Fastrack et Ascension Flock of Birds (Figure 4-1).

Figure 4-1. Capture de mouvements avec le Flock of Birds et reproduction sur un acteur de synthèse

4.2.1.4 Avantages et désavantages des systèmes de capture de mouvement

Prenons, comme exemple, le mouvement de marche humaine qui peut être enregistré et ensuite appliqué à un acteur digital. On aura un excellent mouvement, car il provient directement de la réalité. Cependant, la capture de mouvement n'apporte pas de concept nouveau à la méthodologie de l'animation. Pour un nouveau mouvement, il est nécessaire d'enregistrer de nouveau la réalité. La capture de mouvement n'est donc pas possible dans les cas suivants:

- dans les activités de simulation en temps réel, où la situation et les actions des personnages virtuels ne peuvent pas être prédits à l'avance.
- dans les situations dangereuses, où on ne peut pas impliquer une vraie personne.

4.2.2 L'animation par images-clés

4.2.2.1 Principe

L'animation par images-clés est la méthode la plus simple et la plus primitive. On fournit à l'ordinateur une série d'images à des temps donnés et l'ordinateur calcule les images intermédiaires par interpolation. Cette méthode permet de transformer une forme géométrique en une autre lors d'une animation. La Figure 4-2 montre le principe.

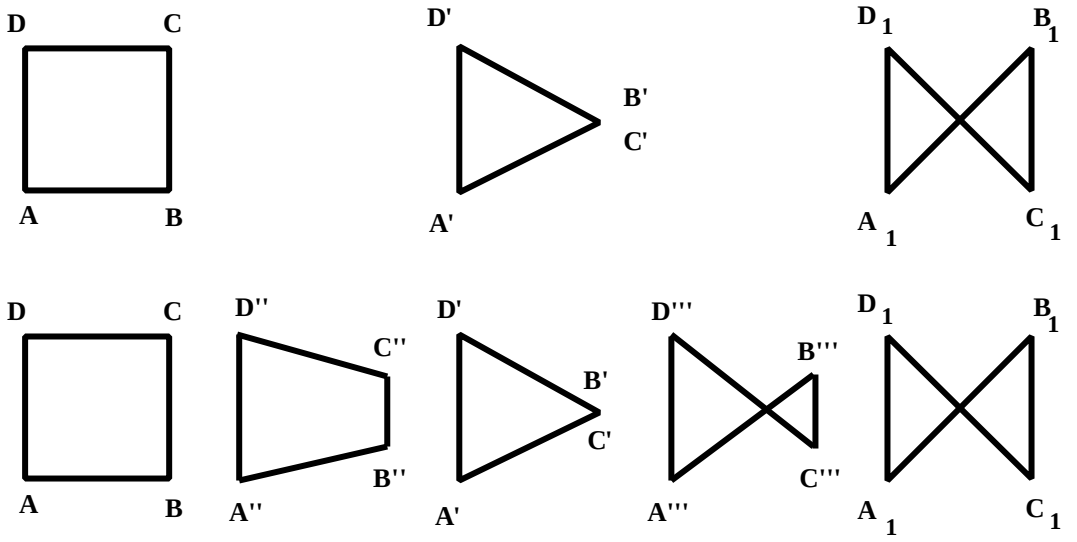


Figure 4-2. Interpolation entre deux dessins; en haut, on montre une interpolation à 50%; en bas, on montre des interpolations à 25%, 50% et 100%.

4.2.2.2 Correspondances

Le problème se complique lorsque les deux dessins n'ont pas le même nombre de sommets. Dans ce cas, il faut d'abord procéder à un prétraitement consistant en l'égalisation du nombre de sommets des deux dessins. Plusieurs algorithmes sont possibles. Le plus simple est le suivant:

> Soient N_1 et N_2 les nombres de sommets des 2 dessins.
> Si $N_1 > N_2$ alors
> on calcule le rapport RT:=(N_1-1) div (N_2-1) et le reste RS:=(N_1-1) mod (N_2-1)
> on ajoute RT points aux RS premiers segments et RT-1 aux autres.

Dans l'exemple de la Figure 4-3, on a N_1=15 et N_2=7; ainsi RS=2 et RT=2. On ajoute donc 2 points aux 2 premiers segments et 1 point aux 4 autres. Ensuite l'interpolation devient simple comme avant.

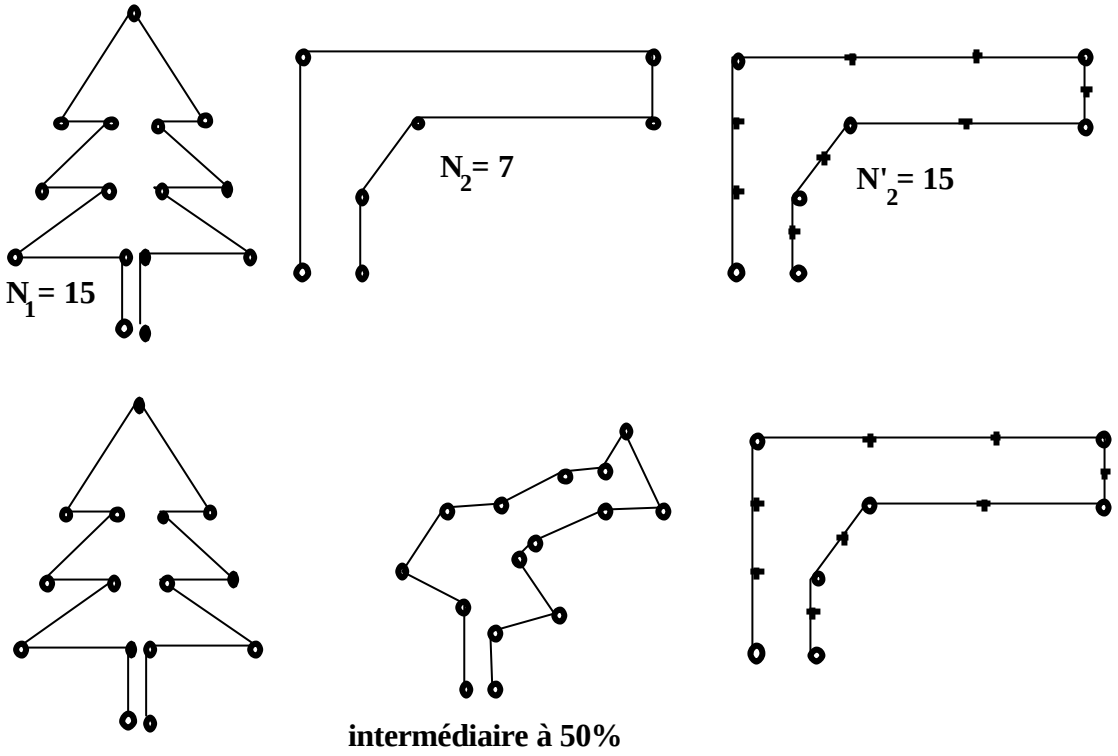


Figure 4-3. Exemple de prétraitement, puis d'interpolation

Cette méthode a été rendue très populaire, particulièrement grâce à l'artiste Peter Foldes, qui a créé de merveilleux films basés sur cette technique: *La Faim* (1974) et *Metadata* (1971). Malheureusement, la technique est ancienne et un algorithme d'interpolation linéaire peut provoquer des effets indésirables tels que le manque de fluidité dans le mouvement, des discontinuités dans la vitesse du mouvement et des distorsions notamment dans les rotations, comme le montre la Figure 4-4.

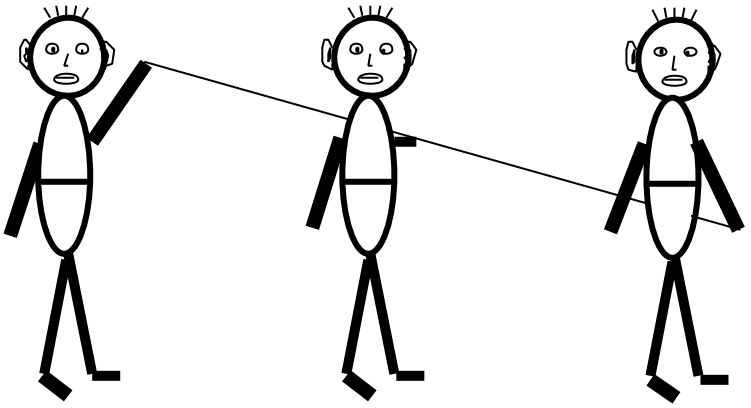


Figure 4-4. Dans cet exemple, on montre que l'interpolation linéaire conduit à un raccourcissement du bras

D'autres méthodes ont été proposées par Baecker [1], Burtnyk and Wein [2], Reeves [3]. Cependant, comme le relèvent Steketee and Badler [4], il n'y a aucune solution totalement satisfaisante aux déviations existant entre l'image interpolée et la modélisation souhaitée.

La méthode de prétraitement peut être étendue aux figures à trois dimensions. Le principe est le même si les figures sont modélisées en lignes. Mais lorsqu'elles sont modélisées en facettes, la technique devient beaucoup plus complexe, car il faut assurer une correspondance entre facettes et sommets. Il faut donc ajouter des sommets et des facettes pour que les 2 figures en aient le même nombre.

4.2.2.3 Un moyen d'éviter les discontinuités temporelles: les P-courbes

La technique est due à Ronald Baecker (1969). Elle définit des courbes décrivant la trajectoire d'un objet et la variation de la vitesse. Dans les P-courbes, les points sont non équidistants dans l'espace, mais dans le temps. Quand deux points sont proches, la vitesse est plus petite que quand ils sont distants, d'où on a un contrôle graphique du mouvement. La Figure 4-5 nous montre les principes.

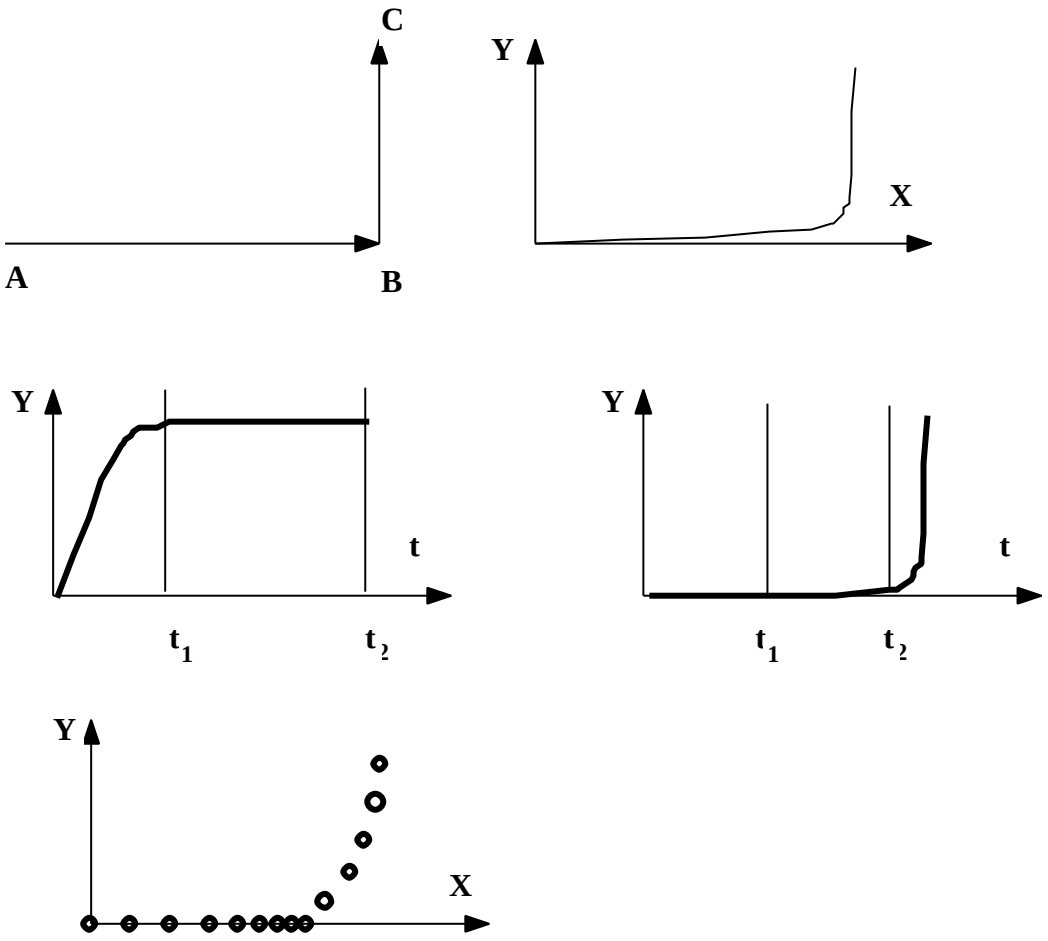


Figure 4-5. Principe des P-courbes. Ici on a une personne qui longe les 2 côtés d'une pièce. On représente successivement la trajectoire y(x), les horaires x(t) et y(t) et enfin la P-courbe.

4.2.2.4 Transformations de formes: "Morphing"

On distingue deux types de morphing [5]: 2D et 3D. Le morphing 2D est une extension de la méthode des images-clés à une interpolation selon les pixels plutôt que les sommets de figures.

Le morphing 3D s'applique à des objets graphiques. La méthode est complexe car le nombre de sommets et de facettes ne correspond généralement pas entre les deux objets. Des sommets et des

facettes doivent être ajoutées de manière à avoir le même nombre pour les deux objets. La Figure 4-6 nous montre un exemple.

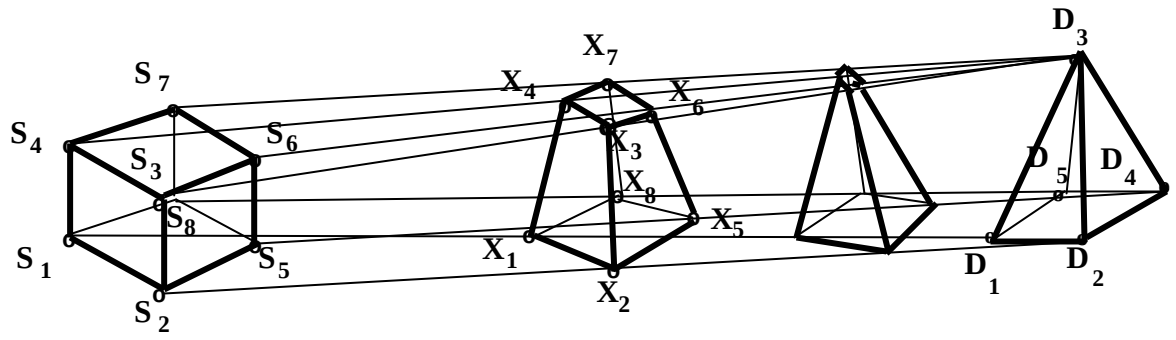


Figure 4-6. Une interpolation 3D

4.2.3 L'animation par interpolation paramétrique

L'animation par interpolation paramétrique est basée sur le principe suivant: on caractérise une entité (objet, caméra, lumière) par des paramètres. On fixe les paramètres à des temps donnés; l'ordinateur calcule les valeurs intermédiaires des paramètres par interpolation et recalcule la scène avec les valeurs interpolées.

Prenons un exemple: l'articulation d'un bras de robot est caractérisée par un angle α variant au cours du temps t; on fixe les valeurs suivantes:

t=0 α=10
t=2 α=20
t=5 α=45
t=8 α=100

Pour connaître la valeur de l'angle tous les 1/25 de seconde, on peut calculer par interpolation linéaire: p.e. pour t = 1/25, on a α= 10+(20-10)/(2*25) = 10.2.

Pourtant, comme le montre la Figure 4-7, on a un problème de continuité au niveau de la dérivée. En effet, les valeurs d'angle aux temps t=2-1/25, t=2 et t=2+1/25 sont respectivement de: α= 20-(20 – 10)/2*25= 19.8; α= 20, α = 20 +(45-20)/(3*25) = 20.33... On a ainsi un saut au voisinage de 2. Donc, l'interpolation linéaire occasionne des discontinuités au niveau de la dérivée, donc de la vitesse et par conséquent, l'animation va être saccadée. On utilisera donc beaucoup plus volontiers une interpolation cubique (Figure 4-8).

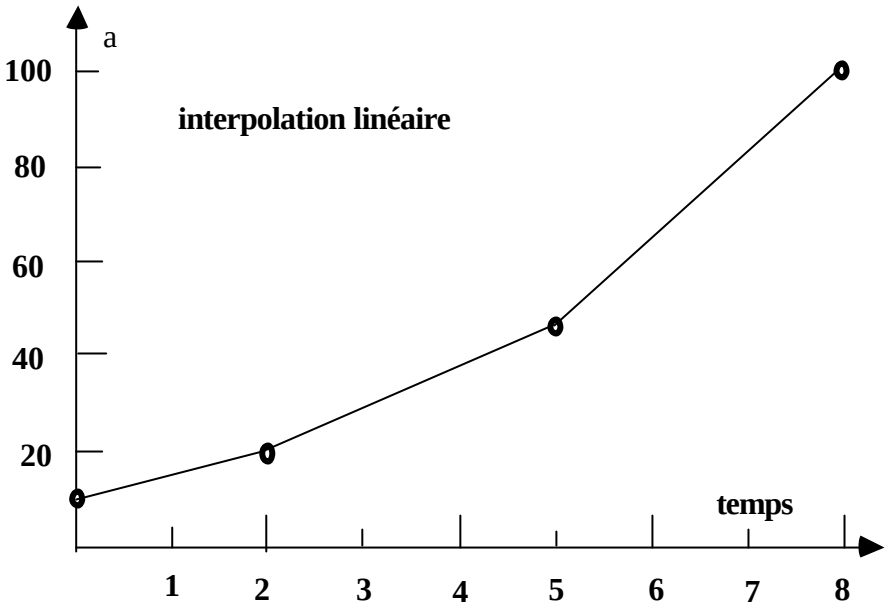


Figure 4-7. Interpolation linéaire de l'angle

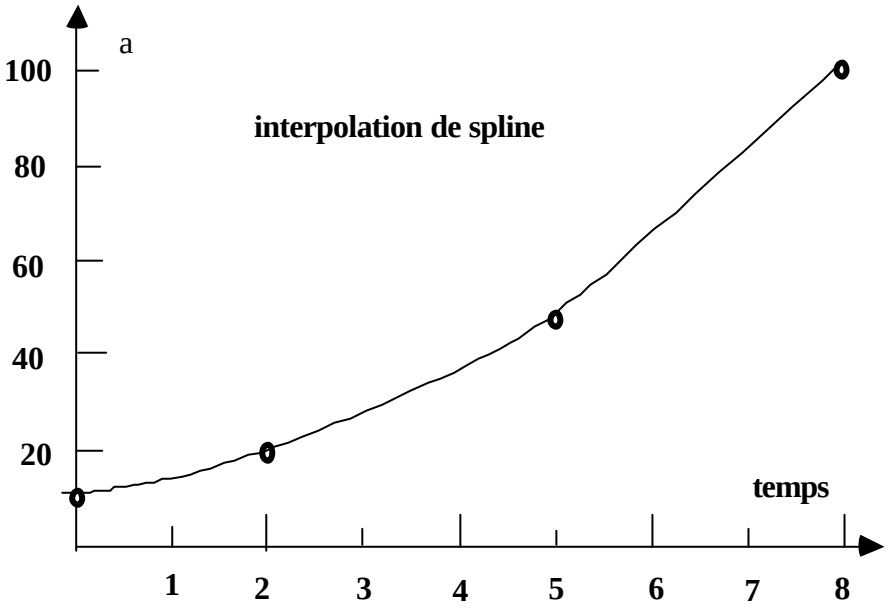


Figure 4-8. Interpolation de spline

Nous allons donc reprendre l'interpolation de spline de Kochanek-Bartels [6] vue à la Section 2.2.7. Le paramètre de tension t contrôle comment la courbe se tend au point P_i. Comme on le voit à la Figure 4-9, dans certains cas une courbe large est souhaitée, alors que dans d'autres cas, on préfère quelque chose de plus brusque.

La continuité c de la spline à un point P_i est contrôlée par le paramètre c. La continuité du mouvement en direction et vitesse n'est pas toujours souhaitée. Ainsi l'animation d'une balle rebondissant sur un mur demande l'introduction d'une discontinuité du mouvement au point d'impact, comme le montre la Figure 4-10. En effet, on ne souhaite pas voir la balle freiner avant de rencontrer le mur.

La direction de la trajectoire lorsqu'elle passe à un point P_i est contrôlée par le paramètre de biais b. Ceci permet à l'animateur d'anticiper ou de retarder une position ou un événement, comme le montre la Figure 4-11.

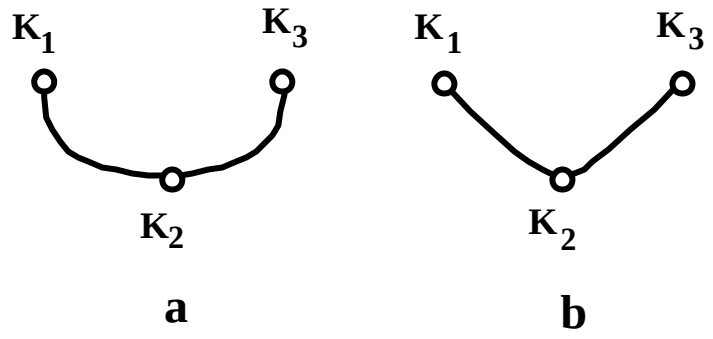


Figure 4-9. Variation de la tension: l'interpolation en b est plus tendue que l'interpolation en a

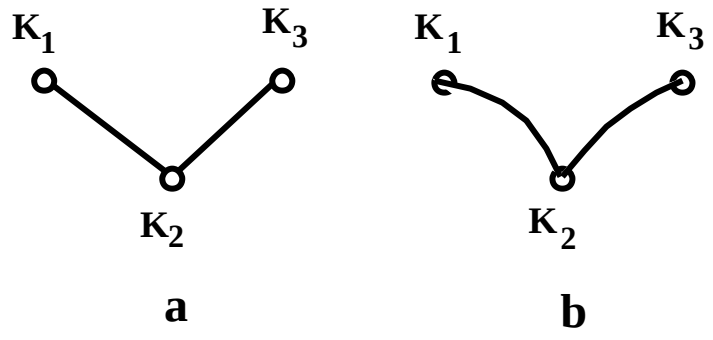


Figure 4-10. Variation de la continuité: l'interpolation en b est plus discontinue que l'interpolation en a

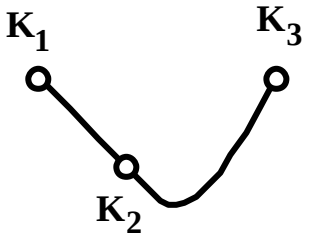


Figure 4-11. Une interpolation biaisée en K2

4.2.4 L'animation procédurale

Dans l'animation procédurale ou algorithmique, le mouvement est décrit algorithmiquement. Prenons un exemple pour comprendre le concept et le comparer avec l'animation par interpolation paramétrique. Nous supposons un objet tombant en chute libre d'une hauteur h=100. Nous désirons

générer la séquence d'animation correspondant au mouvement de l'objet. Avec une approximation de l'accélération de la pesanteur g=10, nous avons la loi $y = 100 - 5\ t^2$. Cela signifie qu'après une seconde, l'objet sera à la hauteur $y1 = 95$, après 2 secondes à la hauteur $y2 = 80$, après 3 secondes à la hauteur $y3 = 55$, après 4 secondes à la hauteur $y4 = 20$. Au temps t=5, l'objet est déjà au sol. Dans un système d'animation par interpolation paramétrique, on devrait fournir à l'ordinateur l'ensemble suivant: {<0,100>, <1,95>, <2,80>, <3,55>, <4,20>} et espérer que la machine générera des images à chaque 1/25 seconde. Par interpolation linéaire, c'est bien sûr irréaliste. En utilisant une interpolation quadratique, le résultat sera exact, puisque la loi physique est de nature quadratique. Cependant, il est plus direct d'utiliser l'équation du mouvement:

```
p.e.    créer OBJET (...);
        TEMPS = 0;
        tantque Y > 0
                Y = 100 - 5*TEMPS^2
                deplacer (OBJET, X,Y,Z);
                dessiner OBJET;
                enregistrer l'image
                effacer OBJET
                TEMPS:=TEMPS+1/25;
```

L'animation algorithmique correspond à une animation basée sur des lois généralement physiques. Par exemple, on fera varier le pendule d'une horloge selon la loi physique du pendule. Ainsi, on aura:

$$\alpha = A \sin(\omega t + \phi)$$

Donc la programmation d'une animation typique peut être de la forme suivante:

```
créer HORLOGE (...);
pour IMAGE:=1 a NB_IMAGES
    TEMPS:=TEMPS+1/25;
    ANGLE:=A*SIN (OMEGA*TEMPS+PHI);
    MODIFIER (HORLOGE, ANGLE);
    dessiner HORLOGE;
    enregistrer l'image
    effacer HORLOGE
```

L'avantage d'utiliser des lois algorithmiques (p.e. physiques) pour l'animation est qu'on obtient un mouvement exact par rapport à la loi. C'est en fait une pure simulation. Les lois de la physique sont appliquées aux paramètres des objets animés (p.e. positions, angles). Le contrôle de ces lois peut être donné par programmation comme dans ASAS [7] et MIRA [8] ou au moyen d'une approche de commandes interactives comme dans le système MIRANIM [9]. Avec une telle approche, n'importe quelle sorte de loi [10] peut être appliquée aux paramètres. Par exemple, la variation d'un angle d'articulation peut être contrôlé par la cinématique aussi bien que par la dynamique (voir Section 4.6).

Les trois méthodes (images-clés, interpolation paramétrique et animation procédurale) peuvent être décrites d'une manière plus uniforme et intégrée. Un objet animé est caractérisé par un ensemble de

variables d'état qui dirigent le mouvement. L'évolution de ces variables d'état est définie par une loi d'évolution. Les trois types d'animation peuvent alors être redéfinis en utilisant la terminologie montrée à la Figure 4-12.

TYPE D'ANIMATION	VARIABLES D'ETAT	LOIS D'EVOLUTION
images-clés	sommets	interpolation linéaire interpolation de spline interpolation de Reeves
interpolation paramétrique	paramètres	interpolation linéaire interpolation de spline
animation procédurale	paramètres	lois physiques

Figure 4-12. Variables d'état et lois d'évolution

4.2.5 L'animation de caméras et de lumières

Une scène d'animation est seulement utile si elle est vue. Mais elle peut être vue de différentes façons. Cela dépend de la position de l'observateur, de la direction de vision et de l'angle de vue. De telles caractéristiques et d'autres sont généralement regroupées dans le concept de caméra virtuelle ou synthétique. Une caméra virtuelle de base est caractérisée par au minimum deux paramètres: l'oeil et le point d'intérêt. L'oeil est un point et représente la position de la caméra; le point d'intérêt est le point vers lequel la caméra est dirigée. Un angle de vue peut aussi être défini pour contrôler l'ouverture de la vision sur la scène. Un des effets les plus spectaculaires de l'animation 3D consiste à se promener à l'intérieur d'une scène donnée. Ces effets sont en fait faciles à produire en animant les caractéristiques de la caméra virtuelle. L'utilisation de plusieurs caméras permet la simulation d'effets spéciaux [11] comme les "fades", "wipes" et "cross-dissolves". Il est aussi essentiel de pouvoir créer des mouvements de caméras selon des trajectoires non-linéaires en utilisant par exemple des splines.

Il est aussi possible de contrôler une caméra virtuelle à l'aide de dispositifs interactifs tels que la SpaceBall ou l'EyePhone. Ainsi Turner et al. [12] ont décrit une interaction basée sur un modèle physique de caméra. L'approche consiste à créer un modèle physique abstrait de caméra utilisant les lois de la mécanique classique pour simuler le mouvement de caméra en temps réel en réponse à des forces captées par un dispositifs 3D comme la SpaceBall.

Une scène doit aussi recevoir de la lumière pour être réaliste. Les lumières synthétiques ont des caractéristiques et celles-ci peuvent évoluer au cours du temps. En particulier, les intensités et les positions des sources de lumière peuvent changer selon des lois d'évolution. Par exemple, nous pouvons définir une source de lumière positionnelle et supposer que l'intensité de la source passe du rouge au vert tandis que la position change selon un mouvement oscillatoire.

4.3 Introduction aux corps articulés et aux acteurs de synthèse

4.3.1 Les acteurs de synthèse

Si l'animateur peut plonger dans son monde virtuel, il voudrait peut être aussi y rencontrer des humains de synthèse. En fait, la recherche dans le domaine de l'animation par ordinateur évolue de plus en plus vers l'animation de scènes comportant des êtres humains **conscients de leur environnement**. Ce type d'animation est **pluridisciplinaire** puisqu'il doit intégrer certains aspects et méthodes spécifiques à l'animation, la mécanique, la robotique, la physiologie, la psychologie et l'intelligence artificielle. Pour créer des mouvements humains raisonnablement naturels, il faut tenir compte à la fois du contexte géométrique, physique et comportemental. Aucun système uniquement basé sur un de ces trois aspects ne peut donner de bons résultats.

L'animation d'un acteur synthétique [13] peut être subdivisée en deux parties: l'animation du corps et celle du visage, car les problèmes à résoudre ne sont pas de même nature. L'animation du corps repose principalement sur des degrés de liberté (angles) tandis que l'animation faciale est avant tout de nature musculaire.

4.3.2 Définition et animation de squelettes

L'animateur spécifie les séquences d'animation pour un acteur en utilisant un squelette. Un squelette est une structure hiérarchique formée d'un ensemble de segments connectés, correspondant aux membres et articulations. Une articulation représente une intersection de deux segments, donc un point du squelette où un membre attaché à ce point peut bouger. L'angle entre deux segments est l'angle d'articulation. Une articulation peut avoir au plus 3 sortes d'angles de position: flexion, pivot et "twisting". La

Figure 4-13 nous montre les différentes articulations du squelette.

La flexion est une rotation du membre influencée par l'articulation et causant le mouvement de tous les membres liés à cette articulation. Cette flexion est faire relativement au point d'articulation et un axe de flexion à définir. Le pivot fait tourner l'axe de flexion autour du membre influencé par l'articulation. Le "twisting": cause la torsion du membre influencé par l'articulation. La direction de l'axe de "twisting" est trouvée de façon semblable à la direction du pivot.

L'animation d'un squelette consiste à animer les angles d'articulation. On observe 2 tendances:

1) mouvement est produit "à la main": capture de mouvement, positions-clés

2) contrôle automatique du mouvement

La première approche est plus populaire pour la production de films par ordinateur.

Nom	Nombre	Angles
VERTEBRE 1	2	FTP
VERTEBRE 2	3	FTP
VERTEBRE 3	4	FTP
VERTEBRE 4	5	FTP
VERTEBRE 5	6	FTP
CLAVICULE GAUCHE	7	FP
CLAVICULE DROITE	11	FP
EPAULE GAUCHE	8	FTP
EPAULE DROITE	12	FTP
COUDE GAUCHE	9	FT
COUDE DROIT	13	FT
POIGNET GAUCHE	10	FP
POIGNET DROIT	14	FP
HANCHE GAUCHE	15	F
HANCHE DROITE	20	F
CUISSE GAUCHE	16	FTP
CUISSE DROITE	21	FTP
GENOU GAUCHE	17	F
GENOU DROITE	22	F
CHEVILLE GAUCHE	18	F
CHEVILLE DROITE	23	F
TALON GAUCHE	19	F
TALON DROIT	24	F

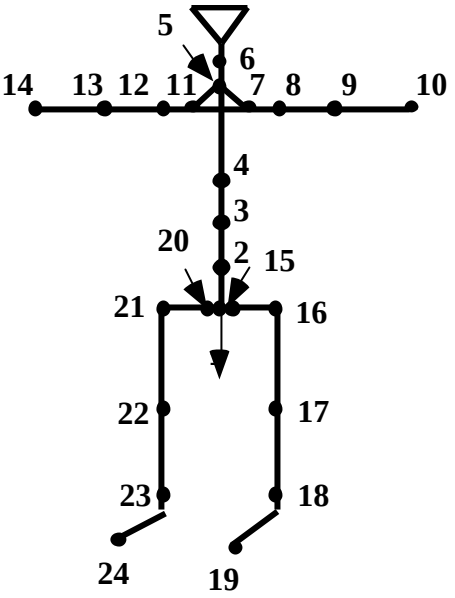


Figure 4-13. Squelette en fil de fer avec les articulations

4.3.3 Mouvement “à la main”

4.3.3.1 Les méthodes

Deux méthodes sont les plus communes dans l’industrie du film:

- La capture de mouvement (voir Section 4.2.1)
- L’animation paramétrique par positions-clés (voir Section 4.2.3)

Dans l'animation par interpolation, on fournit à l'ordinateur un certain nombre de valeurs d'angles (valeurs-clés) à des temps donnés pour les parties du squelette à animer.

Par exemple pour plier un bras, on doit spécifier à l'ordinateur l'angle du coude en plusieurs instants que l'on a choisis. L'ordinateur calcule les valeurs intermédiaires des paramètres par interpolation (par exemple avec l'interpolation de Kochanek-Bartels) et recalcule le personnage avec les valeurs interpolées.

Il faut noter qu’on doit utiliser des **points fixes**. Ce sont des points non animés pendant une séquence d’animation. Par exemple, pour faire s’asseoir une actrice de synthèse, des points fixes doivent être définis aux pieds pour éviter que les jambes ne se soulèvent! Pour une marche, les points fixes doivent changer d’un pied à l’autre.

4.3.4 Contrôle automatique du mouvement

Les deux types d'animation ci-dessus ont leurs avantages et leurs inconvénients, mais ils ont en commun le défaut que l'animateur doit spécifier en détail le mouvement, ce qui devient fastidieux pour simuler de simples actions comme courir ou s'asseoir. De plus, l'interaction avec l'environnement est complètement à la charge de l'animateur, ce qui rend très difficile de tenir compte d'obstacles ou de simuler la préhension.

Une approche plus évoluée consiste à contrôler l'animation à un niveau d'**abstraction fonctionnelle**. Il devient alors possible de diriger l'animation en termes de **buts à atteindre** plutôt que de mouvements détaillés. Nous parlerons alors d'**animation par tâches** et d'**animation comportementale** (voir Section 4.9). Dans les futurs systèmes d'animation basés sur les acteurs de synthèse, le contrôle du mouvement se fera de plus en plus automatiquement reposant sur les techniques d'intelligence artificielle et de robotique.

On relève cinq étapes pour parvenir à un contrôle automatique:

Etape 1: Cinématique inverse et contraintes

Etape 2: Dynamique

Etape 3: Interaction avec l'environnement

Etape 4: Animation de niveau tâche

Etape 5: Animation comportementale et acteurs autonomes

Ces différentes tendances sont discutées plus en détails dans les sections suivantes. Mais, nous allons encore discuter du problème de la modélisation des formes humaines et des déformations du corps lorsque les valeurs des angles d'articulation du squelette changent.

4.4 Déformations du corps

Lorsque le problème de l'animation du corps articulé est résolu, il faut traiter le problème de l'enveloppe, c'est-à-dire la simulation des corps proprement dits. On signalera les travaux basés sur le concept d'**opérateurs de déformations locales** [14], de Free-Form Deformations [15] et de metaballes.

En général, l'animateur doit positionner un personnage standard (le squelette) déjà en mémoire de l'ordinateur et dessiné sous forme de fil de fer à l'intérieur du corps de son personnage à animer. L'animateur doit faire coïncider son personnage (Figure 4-14) selon certains points.

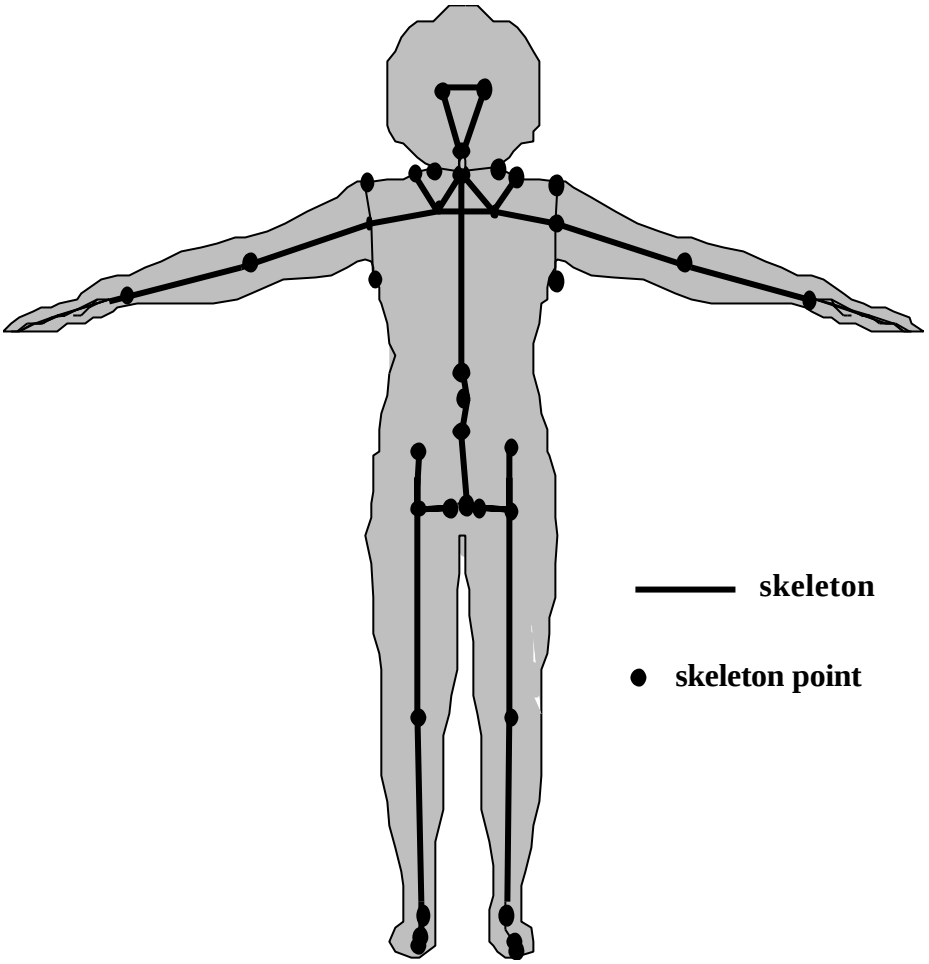


Figure 4-14. Personnage avec points pour les déformations (vue de face)

Cette opération doit être très précise et peut s'avérer assez longue. Elle est cependant nécessaire, car toute l'animation est calculée à partir de ce personnage standard en ligne. En effet, par la suite, le logiciel pourra déformer le personnage selon les angles nécessaires à l'animation sans aucune autre intervention humaine.

Le principe fondamental est le suivant: dissocier l'enveloppement du corps des squelettes. On a alors l'algorithme général de déformation suivant:

pour chaque segment du squelette
    pour chaque point P du corps correspondant au segment
        associer P à l'extrémité la plus proche du segment
        sélectionner un opérateur pour le type d'articulation

Les surfaces recouvrant les articulations sont considérées comme des tuyaux flexibles. Une flexion est appliquée sur l'extrémité du segment correspondant à l'articulation. Les nouvelles coordonnées sont calculées par changement de référentiel.

La méthode développée par Shen [16] au laboratoire d'infographie de l'EPFL est quant à elle basée sur une construction interactive de la structure musculaire et osseuse grâce à un éditeur de metaballes. Ensuite, le système calcule automatiquement des sections par lancer de rayons. Ces

sections servent ensuite à établir une surface B-spline qui correspond à la peau. La Figure 4-15 nous montre le principe. Des exemples sont montrés à la Figure 4-16 et à la Figure 4-17 .

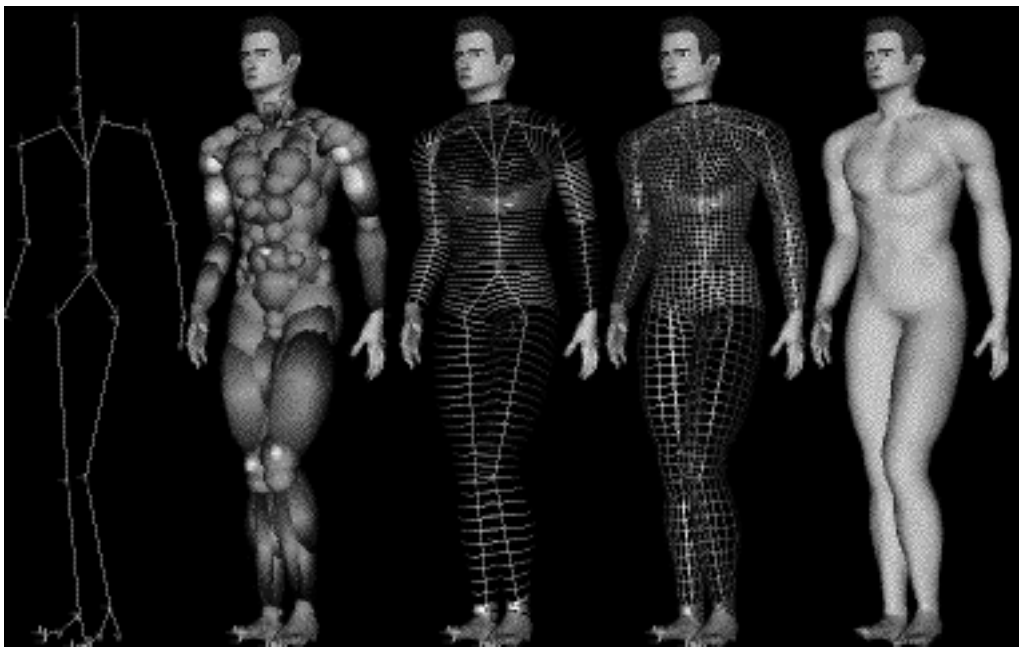


Figure 4-15. Modélisation du corps: squelette, metaballes, sections, lignes, surfaces

4.5 Cinématique directe et inverse

4.5.1 Cinématique directe

On considère une approche basée sur la robotique avec des corps rigides, des mécanismes de liens interconnectés au moyen d'articulations permettant les rotations et éventuellement les translations. Le système d'axe (frame) se déplace avec les liens.

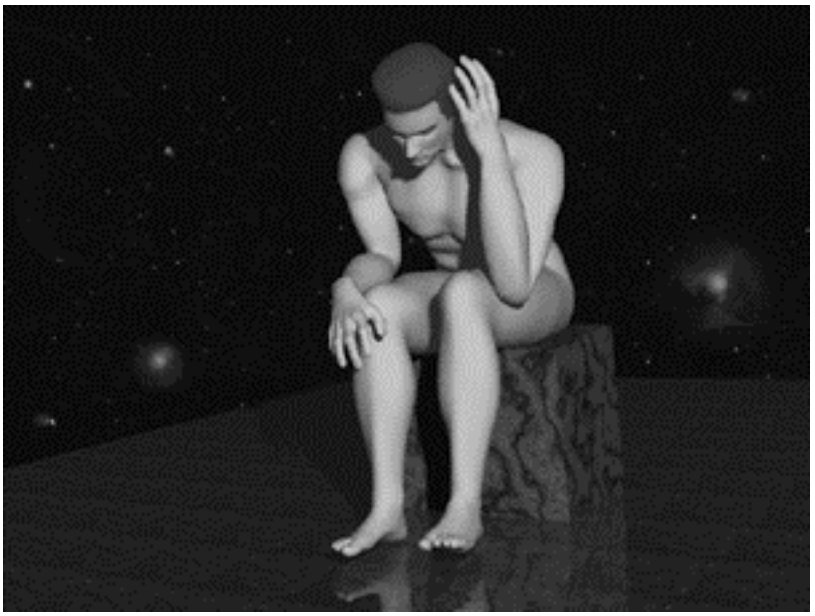


Figure 4-16. Corps d'homme construit avec la méthode des metaballes et des B-splines

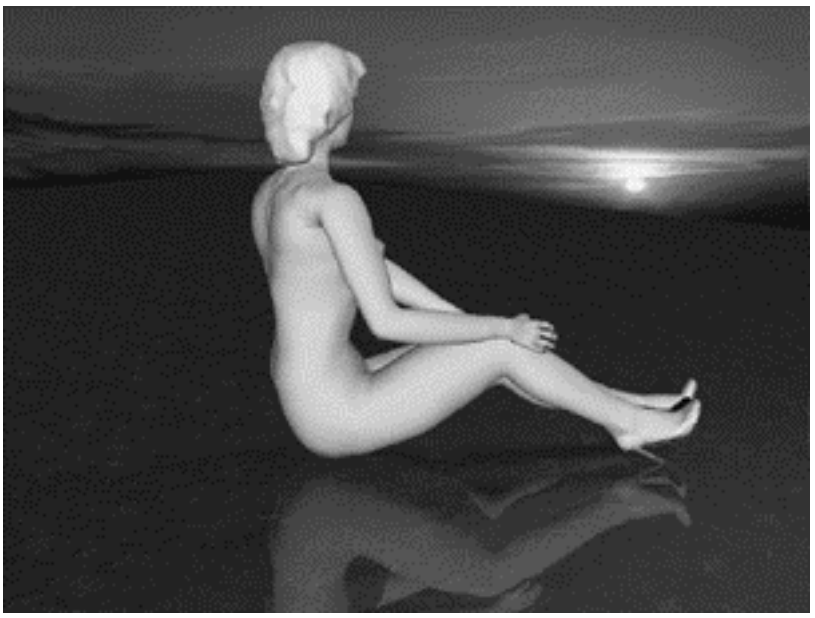


Figure 4-17. Corps de femme construit avec la méthode des metaballes et des B-splines

La méthode d'animation paramétrique est un cas particulier de cinématique directe. Il s'agit de trouver la position et l'orientation d'un manipulateur par rapport à un système de référence comme une fonction du temps sans tenir compte des forces et des moments causant le mouvement. Ceci correspond donc à résoudre l'équation $R = f(\theta)$

$R=(r_x,r_y,r_z,r_\rho,r_\theta,r_\psi)$: position terminatrice
$\theta=(\theta_1,\theta_2,\theta_3,\theta_4,\theta_5,\theta_6)$: vecteur des angles d'articulation.

Des méthodes efficaces et numériquement valables existent. A titre d'exemple considérons le cas d'un manipulateur à 6 degrés de liberté avec 6 liens et 6 articulations ainsi qu'un système de coordonnées (frame) attaché à chaque articulation (voir Figure 4-18).

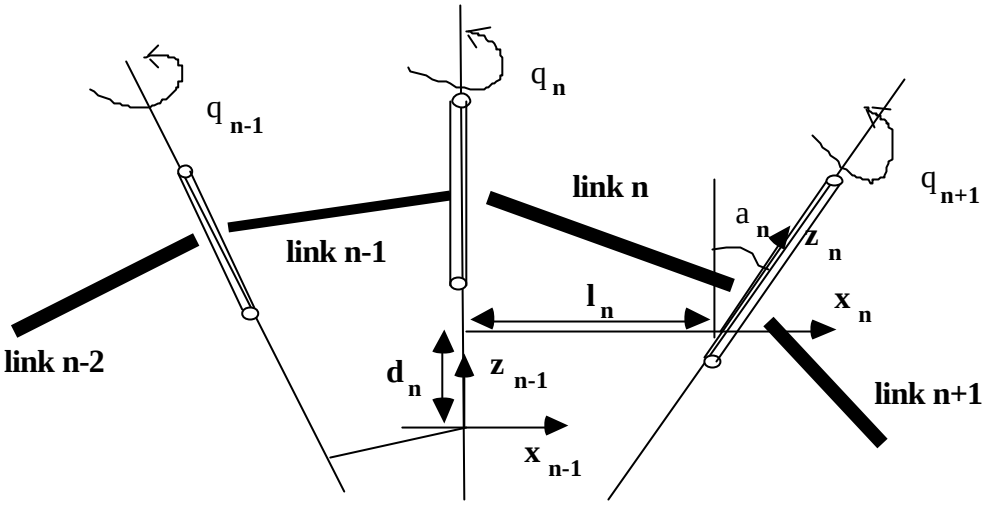


Figure 4-18. Articulations et liens (notations)

Chaque lien k est caractérisé par 4 quantités:

1. longueur l_k: plus courte distance le long de la normale commune aux axes d'articulation.
2. angle de torsion α_k: angle entre les axes dans un plan perpendiculaire à l_k.
3. distance d_k entre les normales le long de l'axe d'articulation.
4. angle d'articulation θ_k.

(l_k, α_k): paramètres de liaisons
(d_k, θ_k): joint paramètres d'articulation.

rotations: l_k,a_k,d_k constants; q_k variable

déplacement: l_k,a_k,q_k constants; d_k variable

On aura les conventions:

- l'axe z_{k-1} se trouve le long de l'axe du mouvement de la k-ième articulation
- l'axe x_k est perpendiculaire à l'axe z_{k-1}, en pointant vers la direction d'éloignement
- l'axe y_k est choisi de manière à avoir un système droit

La relation entre des systèmes de coordonnées successifs (k-1,k) est définie par les 4 transformations:

1. rotation $R_{z\theta}$ de θ_k autour de l'axe z_{k-1} de manière à aligner l'axe x_{k-1} avec l'axe x_k

2. translation T_{zd} de d_k le long de l'axe z_{k-1} pour amener les axes x_{k-1} et x_k à coïncider.
3. translation T_{xl} de l_k le long de l'axe x_k pour amener les 2 origines à coïncider.
4. rotation $R_{x\alpha}$ de α_k autour de l'axe x_k de manière à faire coïncider les 2 systèmes de coordonnées.

On introduit alors la matrice de Denavit-Hartenberg:

$$A_k = R_{z\theta} T_{zd} T_{xl} R_{x\alpha}$$

A_k est une matrice liant les positions et directions dans le lien k au lien k-1:

$$k\text{-}1x = Ak\ kx$$

La position et l'orientation de l'extrémité de la chaîne est alors: $W_6 = A_1 A_2 A_3 A_4 A_5 A_6$

4.5.2 Cinématique inverse et contraintes positionnelles

Il s'agit de contrôler l'animation en termes de positions des membres plutôt qu'en termes de degrés de liberté, c'est-à-dire qu'il s'agit de déterminer les variables d'articulation à partir des positions et orientations de l'extrémité du manipulateur (Figure 4-19).

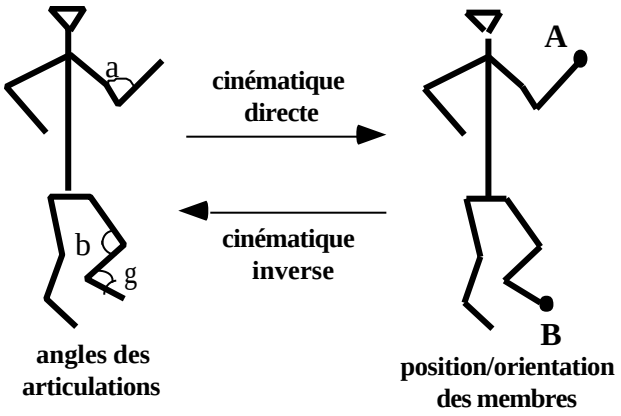


Figure 4-19. Cinématique directe et inverse

Il n'y a pas de solution générale, mais une solution pour des cas limités

p.e. manipulateurs ayant 6 degrés de liberté de rotation, avec trois proches de la terminaison s'intersectant en un point (poignet).

Featherstone [17] a proposé de séparer le problème en 2 sous-problèmes:

1. trouver la valeur des trois premières variables d'articulation afin de positionner le poignet correctement
2. trouver les valeurs des angles du poignet pour positionner la terminaison correctement étant donné l'orientation du poignet calculée en (1)

La cinématique inverse a d'importants défauts. En particulier, elle fonctionne bien pour de simples liens, mais mal pour des trop complexes. Ainsi, il est facile de trouver de combien fléchir un coude et un poignet pour parvenir à un objet avec la main. Il n'est pas si aisé de mettre encore en jeu l'épaule et les doigts.

De plus, il y a le problème du choix lorsqu'il y a plusieurs possibilités et l'animateur doit alors intervenir ou laisser le système faire un choix (par minimisation d'énergie par exemple).

4.5.3 Problème de redondance

Il y a trop de possibilités et l'animateur doit fournir plus d'information. Quelle est la solution la plus naturelle ? Comment spécifier ce choix ? La Figure 4-20 nous montre des choix multiples.

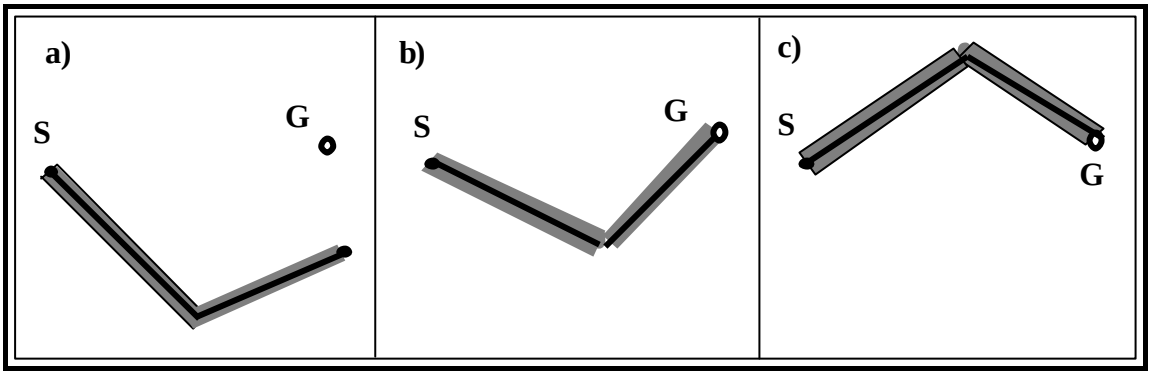


Figure 4-20. Illustration des possibilités multiples: a. conditions initiales. b. première solution. c. seconde solution.

Pour un bras 2D à 3 articulations, il y a au plus 4 solutions. Dans le cas général, avec N degrés de liberté, le nombre de solutions est au plus 2^N. Des solutions peuvent être impossibles à cause d'obstacles comme le montre la Figure 4-21:

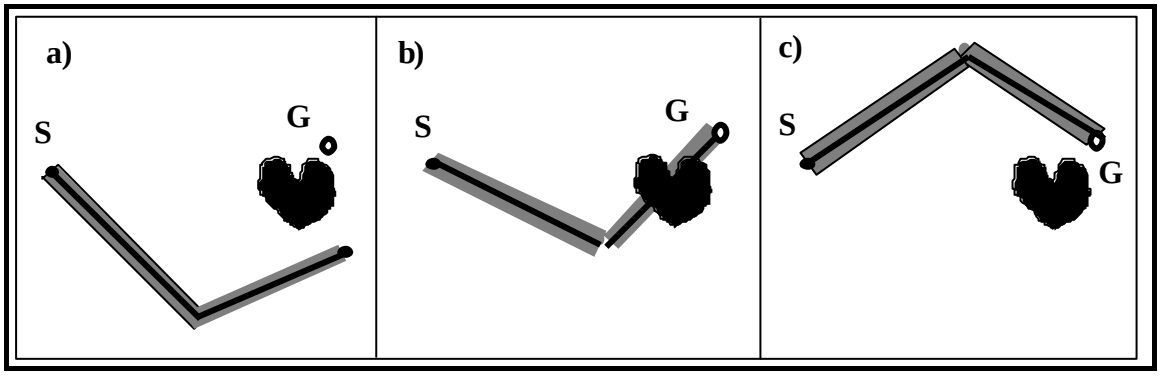


Figure 4-21. Solutions impossibles en cas d'obstacles: a. conditions initiales. b. impossible à cause de l'obstacle. c. solution unique.

4.5.4 Concept du problème d'accès

On définit un espace de travail pour un acteur virtuel, c'est le volume de l'espace que l'extrémité du membre de l'acteur peut atteindre. Le problème d'accès peut seulement être résolu si le but spécifié est dans l'espace de travail.

On a deux types d'espaces de travail:

1. l'espace de travail complet: espace de travail que l'acteur peut atteindre dans toutes les orientations.
2. l'espace de travail que l'acteur peut atteindre dans au moins une orientation.

Exemple: bras simple avec deux segments et deux articulations (Figure 4-22).

On a deux cas:

1º $L_1=L_2$ Espace complet: point S Espace atteignable: disque de rayon $2L_1$.

2º L1?L2 Pas d'espace complet La Figure 4-23 nous montre l'espace atteignable.

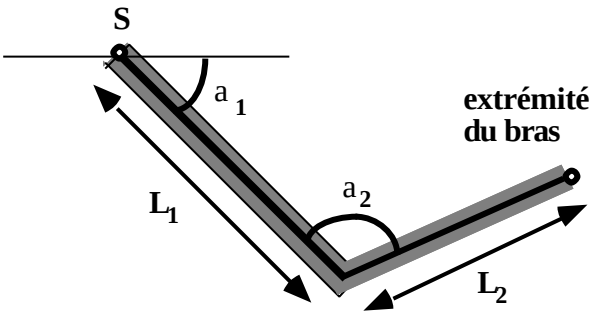


Figure 4-22. Bras simple

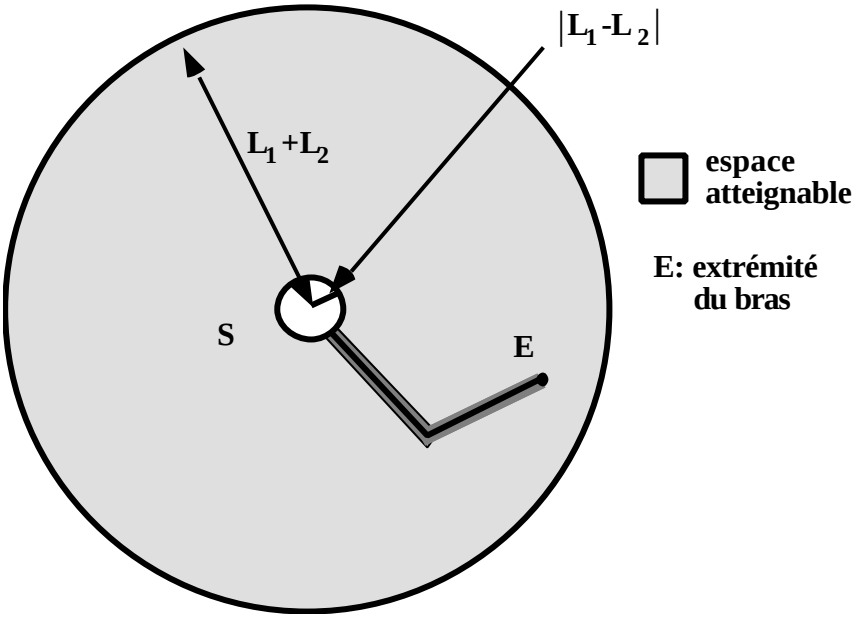


Figure 4-23. Espace atteignable

La possibilité pour un corps articulé d'atteindre des positions et orientations générales dépend du nombre N de degrés de liberté du corps. Si N<6, les corps articulés ne peuvent pas atteindre des buts généraux.

4.5.5 Contraintes cinématiques

Grâce à la cinématique inverse, on peut introduire des contraintes cinématiques. Pour faire asseoir une actrice de synthèse sur une chaise, par exemple, il est nécessaire de spécifier des contraintes adéquates sur les pieds, le bassin et les mains. On peut introduire [18] un algorithme permettant de définir deux types de contraintes:

1. contrainte fixe sous forme d'une position 3D et de trois angles décrivant les orientations
2. contrainte sous forme d'une trajectoire en six dimensions (3 pour les positions et 3 pour les orientations).

Un système qui ne permet de spécifier qu'une contrainte à la fois n'est pas efficace pour résoudre le problème. D'où l'introduction d'algorithme itératif [19] pour résoudre les contraintes multiples avec la cinématique inverse. L'usager doit généralement spécifier la précédence de chaque contrainte pour le cas ou elle ne peuvent pas être toutes satisfaites simultanément.

4.6 Simulations dynamiques

4.6.1 Principes de la dynamique

Les systèmes basés sur la cinématique sont généralement intuitifs et manquent d'intégrité dynamique. L'animation ne semble pas correspondre à des faits physiques communs comme la gravité ou l'inertie. Seuls des objets dont les mouvements sont effectués sous l'action de forces et de moments de forces peuvent être réalistes. Les forces et les moments de forces provoquent des accélérations

linéaires et angulaires. Le mouvement est obtenu par les équations de la dynamique du mouvement. Ces équations sont établies en utilisant les forces, les moments de forces, les contraintes et les propriétés de masse des objets.

Un exemple typique est le mouvement d'un corps articulé subissant des forces et des moments de forces à ses membres. Ces forces et moments de forces peuvent être de différents types:

- moments de forces provenant des liens hiérarchiques parents et enfants,
- forces aux pivots,
- effets externes tels que les contacts avec des objets ou des torsions de bras par exemple

Le contrôle de l'animation est fait en faisant intervenir les **forces** et les **moments de forces** comme causes du mouvement. Les avantages d'utiliser cette dynamique sont les suivants [20]:

- la dynamique libère l'animateur de la description des mouvements dus aux lois physiques
- l'animation des phénomènes est plus réaliste du moins du point de vue physique.
- les corps peuvent réagir automatiquement à des contraintes internes et externes telles que des champs, des collisions, des forces ou des mouvements.

Il y a aussi de sérieux désavantages:

- Les systèmes basés sur la dynamique sont généralement difficile à utiliser pour un animateur.
- Les paramètres (forces, moments de forces) sont parfois très difficiles à ajuster, car ils ne sont pas de nature intuitive.
- Le temps CPU nécessaire pour résoudre les équations du mouvement d'un corps articulé complexe à l'aide de méthodes numériques est souvent prohibitif. Ceci réduit considérablement les possibilités interactives de tels systèmes.

Bien que les mouvements basés sur la dynamique soient plus réalistes, ils sont trop réguliers, car ils ne tiennent pas compte de la personnalité des personnages.

La Figure 4-24 nous montre un exemple typique de mouvement où la dynamique est importante.

Les méthodes basées sur des ajustements de paramètres sont les plus populaires en animation basée sur la dynamique et correspondent aux *méthodes sans contraintes*. Elle seront présentées à la section suivante. Il y a une alternative: les méthodes basées sur des contraintes: l'animateur spécifie en termes de contraintes les propriétés que son modèle est supposé d'avoir, sans avoir à ajuster les paramètres pour lui donner ces propriétés. 4 types de contraintes sont discutées à la section 2.2.6: les *contraintes cinématiques*, les *contraintes dynamiques*, les *contraintes énergétiques* et les *contraintes espace-temps*.

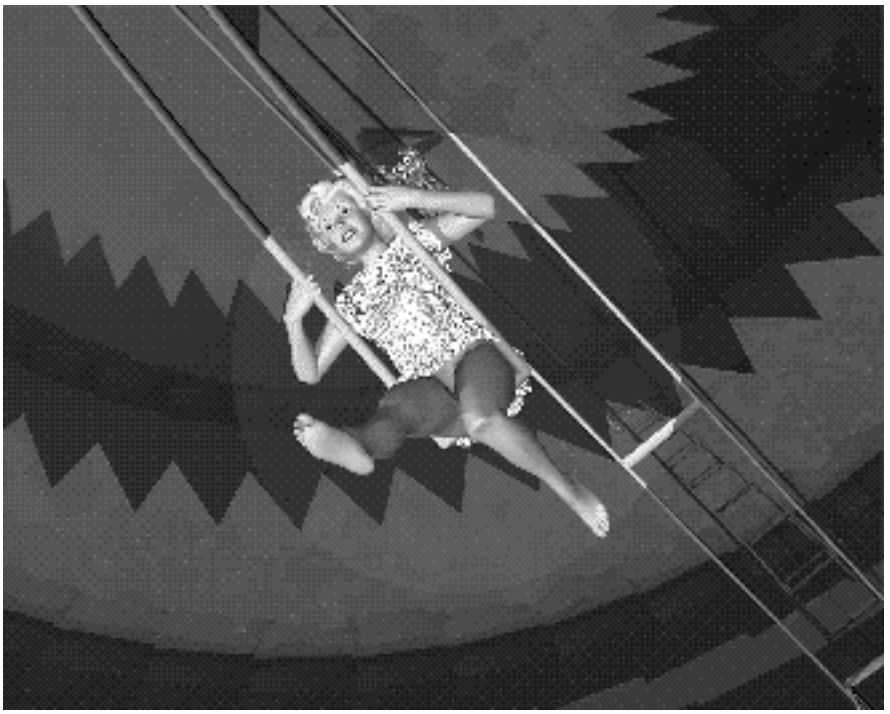


Figure 4-24. Mouvement basé sur la dynamique

En animation basée sur la dynamique, il y deux problèmes à considérer: le problème de la dynamique directe et celui de la dynamique inverse (Figure 4-25). Le problème de la dynamique directe consiste à trouver les trajectoires de points (p.e. une extrémité d'un membre) connaissant les forces et les mouvements qui causent le mouvement. Le problème de la dynamique inverse est généralement plus utile et peut être formulé ainsi: déterminer les forces et les moments de forces nécessaires pour produire un mouvement prédéterminé. Pour une figure articulée, il est possible de calculer la séquence temporelle des moments appliqués aux articulations nécessaire pour exécuter la séquence temporelle désirée de positions, vitesses et accélérations selon plusieurs méthodes.

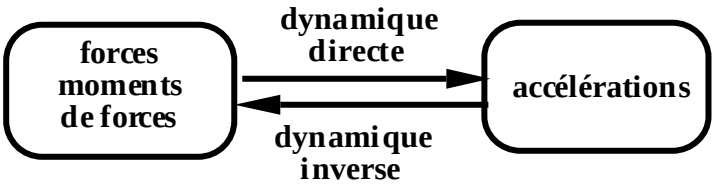


Figure 4-25. Dynamique directe et inverse

4.6.2 Les méthodes sans contraintes

Les méthodes sans contraintes ont été principalement utilisées pour l'animation de figures articulées. Il y a un certain nombre de formulation équivalentes qui utilisent des équations de mouvement différentes:

- la formulation de Newton–Euler
- la formulation de Lagrange
- la formulation de Gibbs–Appell
- la formulation de D'Alembert

Ces formulations sont populaires en robotique et plus de détails sur les équations et leur utilisation en animation par ordinateur peuvent être trouvés dans [21].

4.6.2.1 La formulation de Newton–Euler

La formulation de Newton-Euler est basée sur les lois gouvernant la dynamique des corps rigides. La procédure est la suivante:

- Ecrire les équations définissant les vitesses et accélérations angulaires et linéaires de chaque segment
- Ecrire les équations décrivant les forces et les moments de force exercés sur les segments successifs.

Equation de Newton: $F = m\,a$ (Eq. 4-1)

Equation d'Euler: $N = J\,\alpha + \omega \times (J.\omega)$ (Eq. 4-2)

F: vecteur de la force totale agissant sur le corps de masse m

N: vecteur du moment de force total agissant sur le corps avec une matrice d'inertie autour de son centre de masse.

a: accélération linéaire a

α: accélération angulaire

ω: vitesse angulaire

L'équation (Eq. 4-1) décrit le mouvement linéaire du corps rigide tandis que l'équation (Eq. 4-2) gouverne la rotation du corps rigide.

Les méthodes de Newton-Euler ont été tout d'abord développées avec des systèmes de coordonnées fixes [22]. Puis des méthodes ont été introduites avec des systèmes de coordonnées locaux. Orin et al. [23] ont proposé que les forces et les moments de forces soient exprimées par rapport au système de coordonnées du lien. Armstrong [24] et Luh et al. [25] ont calculé les vitesses angulaires et linéaires dans les coordonnées du lien également.

4.6.2.2 La formulation de Lagrange

Les équations de mouvement pour les robots peuvent être dérivés à partir de l'équation de Lagrange. Uicker [26] et Kahn [27] utilise les matrices 4x4 de rotation/translation introduites par Denavit et Hartenberg. Une fois les énergies cinétique et potentielle exprimées en termes de ces matrices et de leurs dérivées, l'équation de Lagrange est facilement appliquée et les forces généralisées trouvées. Malheureusement, l'évaluation de l'équation est très coûteuse en temps CPU, car elle est proportionnelle à la puissance 4 du nombre de liens dans un corps articulé. Hollerbach [28] a proposé une méthode récursive qui réduit significativement le nombre d'opérations. De manière similaire, Armstrong et al. [29] ont conçu un système basé sur la dynamique et pratiquement en temps réel. Pour réduire le temps CPU, ils font des hypothèses simplificatrices sur la structure des figures.

4.6.2.3 La formulation de Gibbs–Appell

Wilhelms et Barsky [30] utilisent la formulation de Gibbs–Appel pour leur système d'animation Deva; cependant, la méthode a été abandonnée, car les matrices étaient trop importantes et le coût de résolution des équations trop important ($O(n^4)$).

4.6.2.4 La formulation de D'Alembert

Le *principe de D'Alembert des travaux virtuels* affirme que si un système est en équilibre dynamique et les corps sont susceptibles de se déplacer un peu (petits mouvements) alors la somme des travaux des forces appliquées et des forces internes va être égale et opposée au travail occasionné par le changement. C'est sur ce principe qu'une séquence d'animation basée sur la dynamique [31] a été produite, consistant en une actrice (Marilyn) dessinant les lettres O et M avec son bras. Pour produire cette séquence, une approche mécanique a donc été utilisée qui traite des chaînes ouvertes et fermées indifféremment.

4.6.3 Les méthodes basées sur les contraintes

4.6.3.1 Contraintes cinématiques

Isaacs et Cohen [32] ont présenté une méthode de simulation par contraintes basée sur une formulation matricielle. Les articulations sont configurées comme des contraintes cinématiques, et soit les accélérations, soit les forces peuvent être spécifiées pour les liens.

Trois moyens de contrôle ont été définis:

1. contraintes cinématiques permettant aux systèmes traditionnels d'animation par positions-clés d'être incorporés dans une analyse dynamique
2. fonctions de comportement permettant au personnage de réagir à l'environnement
3. dynamique inverse pour déterminer les forces nécessaires à l'accomplissement d'un mouvement donné

4.6.3.2 Contraintes énergétiques

Plus généralement, une approche pour imposer et résoudre des contraintes géométriques sur des modèles paramétrisés a été introduite par Witkin et al. [33]. Les contraintes sont exprimées comme des fonctions d'énergie.

4.6.3.3 Contraintes dynamiques

A l'aide de contraintes dynamiques, Barzel et Barr [34] construisent des objets en spécifiant des contraintes géométriques; les modèles s'assemblent d'eux-mêmes simplement par leurs éléments se déplaçant selon les contraintes. Une fois un modèle construit, il tient grâce aux forces de contraintes. Ces forces sont calculées en les appliquant aux corps de telle manière qu'elles agissent conformément aux contraintes géométriques données par l'usager. Platt et Barr [35] ont étendu le concept de contraintes dynamiques à des modèles flexibles. Ils ont introduit deux types nouveaux de contraintes: les contraintes de réaction et celles d'optimisation.

4.6.3.4 Contraintes espace-temps

Witkin et Kass [36] ont proposé une nouvelle méthode, appelée contraintes espace-temps, pour animer des personnages. Dans cette nouvelle approche, le mouvement du personnage est automatiquement créé en spécifiant ce que le personnage doit faire, comment le mouvement doit être exécuté, quelle est la structure physique du personnage, quelles sont les ressources disponible au personnage pour accomplir le mouvement. Le problème à résoudre est alors un problème d'optimisation sous contraintes. La solution est un mouvement physiquement valide satisfaisant les contraintes et optimisant le critère *comment*. L'approche contrainte espace-temps est conforme aux principes de l'animation traditionnelle comme l'anticipation ou les effets squash-and-stretch.

4.6.4 Etude de cas: la dynamique dans le système TRACK

L'humain est considéré comme un système qui utilise les muscles pour convertir l'énergie emmagasinée en forces et moments de force variant au cours du temps et agissant sur l'articulation. Ce n'est pas un objet passif comme une simple chaîne articulée qui n'est commandée que par des forces et des moments externes.

Problème: comment trouver les forces et moments variant au cours du temps avant d'utiliser la dynamique directe ?

4.6.4.1 Modélisation pour la dynamique

Le squelette humain dans le système TRACK [37] compte environ 206 os au total. Notre modèle (Figure 4-26) est un squelette simplifié avec des figures rigides articulées connectées par un à trois degrés de liberté à chaque articulation. Il contient 49 articulations avec 88 degrés de liberté. On a seulement 4 segments pour la colonne vertébrale (on néglige les vertèbres au niveau du thorax totalement et celles au niveau de l'abdomen partiellement)

Pour la simulation dynamique, le volume de l'humain doit être modélisé. On peut approximer le corps avec 15 primitives solides; cylindres, sphères, ellipsoïdes et cônes tronqués. Il est facile de calculer les propriétés géométriques et physiques avec cette approximation.

La masse de chaque volume est dérivée d'expériences biomécaniques. Le squelette est associé avec des solides, les masses sont listées à la Figure 4-27.

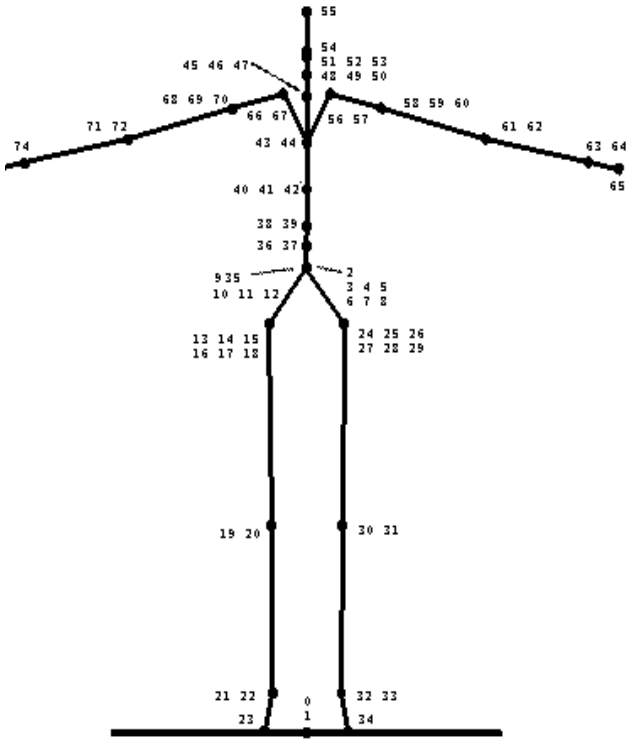
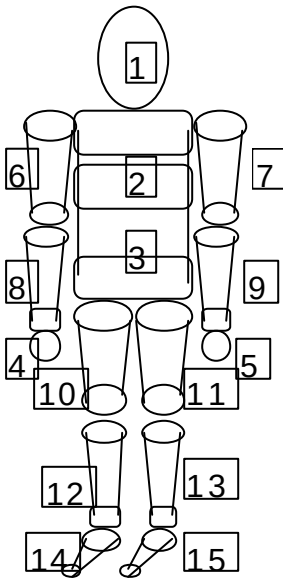


Figure 4-26. Squelette



Nom	Type	Masse (Segment /Total)
tête et cou	ellipsoïde	0.081
up_torso	cylindre	0.216
down_torso	cylindre	0.281
r_hand	sphère	0.006
l_hand	sphère	0.006
r_upper_arm	cône	0.028
l_upper_arm	cône	0.028
r_forearm	cône	0.016
l_forearm	cône	0.016
r_thigh	cône	0.100
l_thigh	cône	0.100
r_leg	cône	0.0465
l_leg	cône	0.0465
r_foot	cône	0.0145
l_foot	cône	0.0145

Figure 4-27. Masses pour le calcul dynamique

Toutes les primitives solides sont considérées comme rigides pour les calculs de dynamique. Elles sont liées par des articulations dans la structure hiérarchique. Le mouvement peut être décrit par la position et l'orientation de chaque solide. Pour décrire le mouvement, nous définissons un système de coordonnées pour chacun d'eux avec l'origine au point charnière avec l'autre solide. Le système de coordonnées du corps se déplace avec le solide. Toutes les propriétés géométriques et physiques définies dans ce système de coordonnées ne changent pas à cause de la rigidité du solide. Pour cette

raison, la plupart des formules sont représentées dans le système de coordonnées du corps pour des raisons de simplicité.

L'avantage d'une connexion avec un seul système de coordonnées, c'est la simplicité quand on veut mettre à jour la structure pendant la simulation dynamique, mais il est très difficile de représenter des nombres différents de degrés de liberté à chaque articulation et des intervalles de valeurs pour chaque degré de liberté.

Dans notre modélisation du corps humain, entre deux segments, on a de un à trois systèmes de coordonnées (degrés de liberté). Un segment associé peut seulement tourner autour de l'axe Z du système de coordonnées. Cela augmente la complexité pour mettre à jour la structure. La Figure 4-28 donne le détail de notre modèle, du thorax au bras droit.

Il y a un problème majeur en contrôle du mouvement, c'est comment obtenir la valeur (variable au cours du temps) du moment de force produit par le muscle à l'articulation donnée pour effectuer le mouvement désiré. On va utiliser la formulation de dynamique inverse basée sur l'équation de Newton-Euler pour obtenir les valeurs de force et de moment (variables dans le temps) pour la simulation en dynamique directe.

4.6.4.2 Simulation en dynamique directe

L'algorithme le plus populaire est l'algorithme d'Armstrong-Green, basé sur les formulations de Newton-Euler. Pour obtenir une forme simplifiée qui nous permette d'éviter une inversion de matrices de dimensions plus grandes que 3, deux hypothèses doivent être faites.

1. on assume une relation linéaire entre l'accélération linéaire du segment et la valeur d'accélération angulaire induite.

2. on assume une relation linéaire entre l'accélération linéaire et la réaction sur le segment parent.

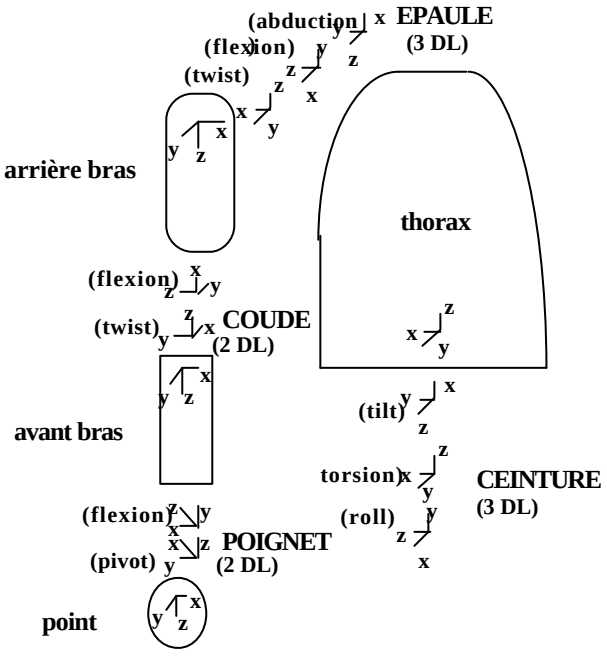


Figure 4-28. Du thorax au bras droit

L'algorithme peut être divisé en deux processus opposés, ascendant et descendant.

- processus ascendant: il calcule des matrices et vecteurs à partir de la structure géométrique et physique du système et propage la force et le moment le long de chaque segment à partir des feuilles jusqu'à la racine de la hiérarchie.

- processus descendant: il calcule les quantités cinématiques, les accélérations linéaire et angulaire de chaque segment, puis obtient les vitesses linéaire et angulaire par intégration numérique afin de mettre à jour la structure entière.

C'est un algorithme récursif qui se déroule tout le temps de la simulation dynamique. Les résultats cinématiques pour un temps sont utilisés comme valeurs initiales pour le calcul au temps suivant.

C'est la méthode d'Euler qui est utilisée par Armstrong-Green pour l'intégration numérique à chaque pas d'où les erreurs s'accumulent d'un pas à l'autre quand on simule le mouvement sur un temps long et on obtient un mouvement non réaliste. On utilise donc la méthode de Runge-Kutta d'ordre 4 pour l'intégration numérique.

Pour obtenir la valeur de vitesse angulaire à chaque pas, on utilise le processus ascendant-descendant quatre fois pour calculer le changement de la vitesse angulaire d'où la stabilité de l'algorithme est améliorée.

Pour trouver les force et moment d'activation de l'articulation pour la simulation en dynamique directe, nous utilisons les formulations de la dynamique inverse.

4.6.4.3 Dynamique inverse

Nous utilisons une sorte de formulation récursive basée sur les équations de Newton-Euler à cause de leur efficacité du point de vue calcul. L'avantage est que le temps de calcul croît linéairement avec le nombre d'articulations.

Pour construire les formulations pour le modèle de corps humain, nous utilisons la même approche qu'en robotique, on écrit une série d'équations pour chaque segment, utilisant des forces et des moments de contraintes pour garantir leur connexions. Nous avons deux processus opposés:

1. récursion vers l'avant: on part du système de coordonnées inertiel vers le système de coordonnées de l'extrémité pour propager l'information cinématique.

2. récursion vers l'arrière: on propage les forces et les moments de force exercés sur chaque segment de la direction opposée.

4.6.4.4 Contrôle avec dynamique inverse

L'utilisation de la dynamique inverse en étroite relation avec la dynamique directe est un moyen clé d'obtenir le résultat souhaité.

A la Figure 4-29, le mouvement désiré est représenté par les variables d'articulation $q, \dot{q}, \ddot{q}$. ?t est le pas de temps pour la dynamique inverse et ?t' est le pas de temps pour la dynamique directe.

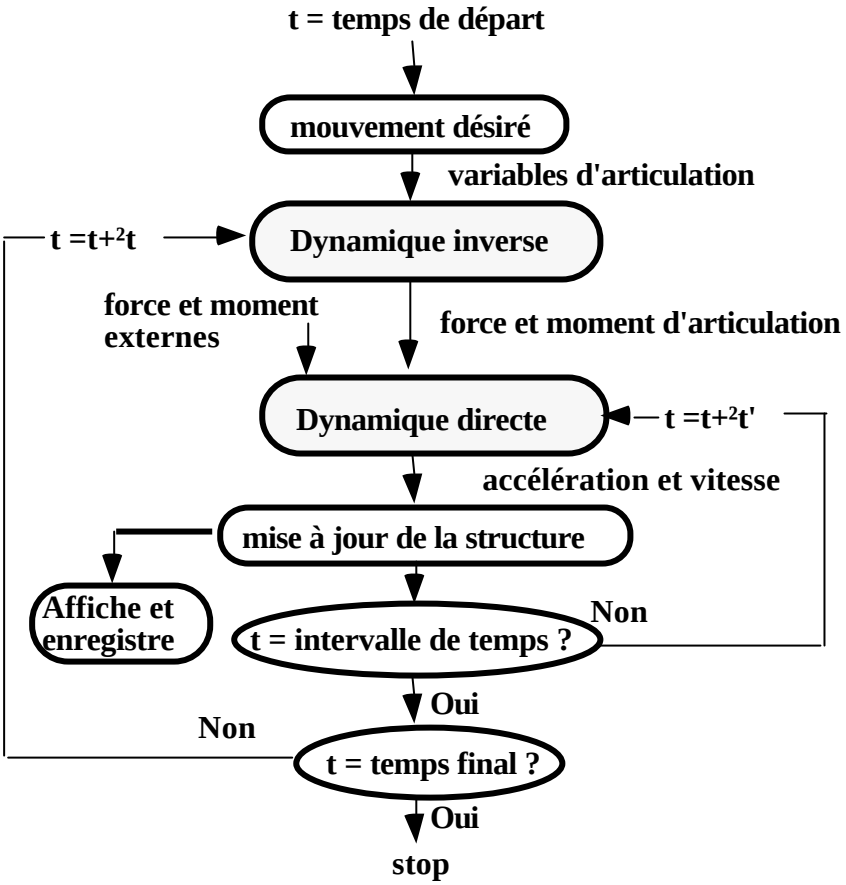


Figure 4-29. Contrôle avec dynamique inverse

4.7 Impact de l'environnement

Il contribue au contrôle adaptatif du mouvement de l'acteur. L'environnement a un impact sur le mouvement de l'acteur et inversement. Les informations sur l'environnement et l'acteur doivent être disponibles pendant le processus de contrôle. L'avantage est que cela décroît le montant d'information introduit dans l'ordinateur par l'animateur. Le système devrait aussi avoir une représentation efficace de la géométrie des objets de façon à prévenir automatiquement les collisions.

4.7.1 Planification de trajectoires

Le problème de la planification de trajectoires est un problème classique étudié en robotique et I.A.

Exemple: soit la position de départ de la main d'un acteur virtuel et la position d'objets sur la table, le problème est de trouver la trajectoire à suivre afin de saisir un objet en évitant les obstacles.

Pour l'acteur, le problème est très complexe si on tient compte de la non-rigidité de l'acteur.

Pour la planification de la trajectoire, il y a deux problèmes de base:

- la détection de la collision

- l'évitement des obstacles

La trajectoire résultant est alors utilisée comme donnée d'entrée au générateur de mouvement:

p.e. résolution de contraintes positionnelles basée sur la cinématique.

4.7.2 Evitement d'obstacles

C'est un problème de planification automatique de chemin. Cela revient à calculer, pour un acteur donné, un chemin libre de toute collision en n'ayant que la destination finale comme donnée. En robotique, c'est un problème classique

4.7.2.1 Première classe d'algorithmes: essais-erreurs

On calcule le volume engendré par le déplacement de l'objet mobile le long du chemin. On détermine l'intersection entre le volume engendré et l'obstacle. Si nécessaire, un nouveau chemin est proposé. En fait, la méthode ne marche que si les obstacles sont rares et disséminées de manière à pouvoir être traités un à la fois.

4.7.2.2 Seconde Classe: algorithmes basés sur des fonctions de pénalité

Une fonction de pénalité encode la présence d'objets. Le mouvement est planifié selon les minima locaux de la fonction de pénalité. On a comme limitation que la fonction de pénalité ne peut être évaluée que pour des formes simples comme des robots sphériques par exemple.

4.7.2.3 Troisième Classe: algorithmes d'espace libre

Un cas typique est l'algorithme du graphe de visibilité de Lozano-Perez [38]. On connecte les sommets qui se voient (Figure 4-30).

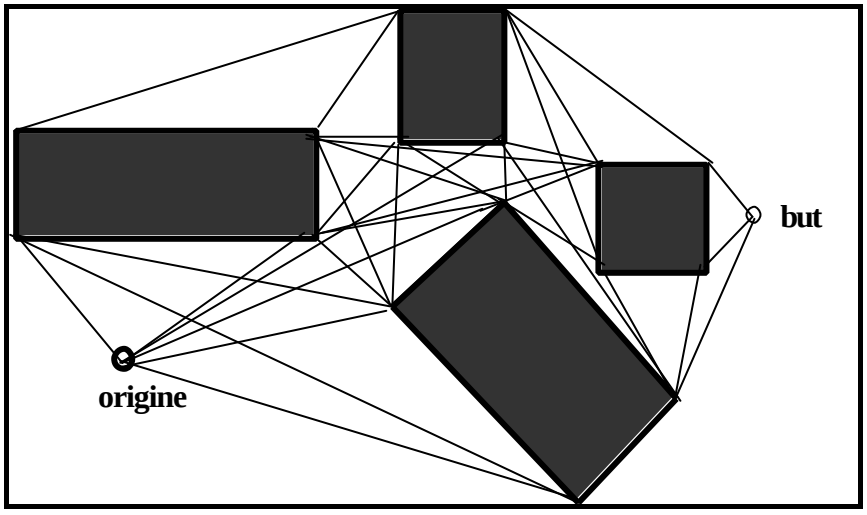


Figure 4-30. Graphe de visibilité

Le chemin le plus court libre de collisions de l'origine au but est le chemin le plus court dans le graphe qui va de l'origine au but (Figure 4-31).

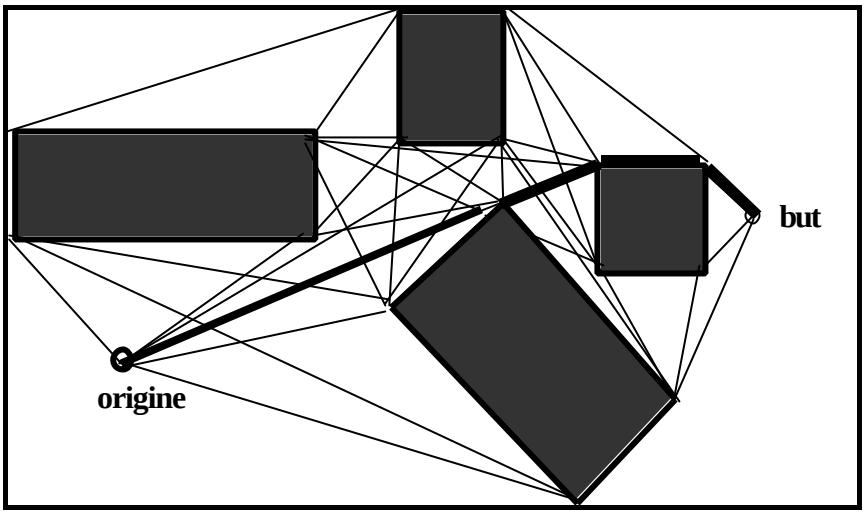


Figure 4-31. Chemin le plus court libre de collisions

Pour étendre l'algorithme aux objets mobiles non ponctuels, les obstacles sont remplacés par de nouveaux obstacles (Figure 4-32), régions interdites pour un point de référence sur l'objet.

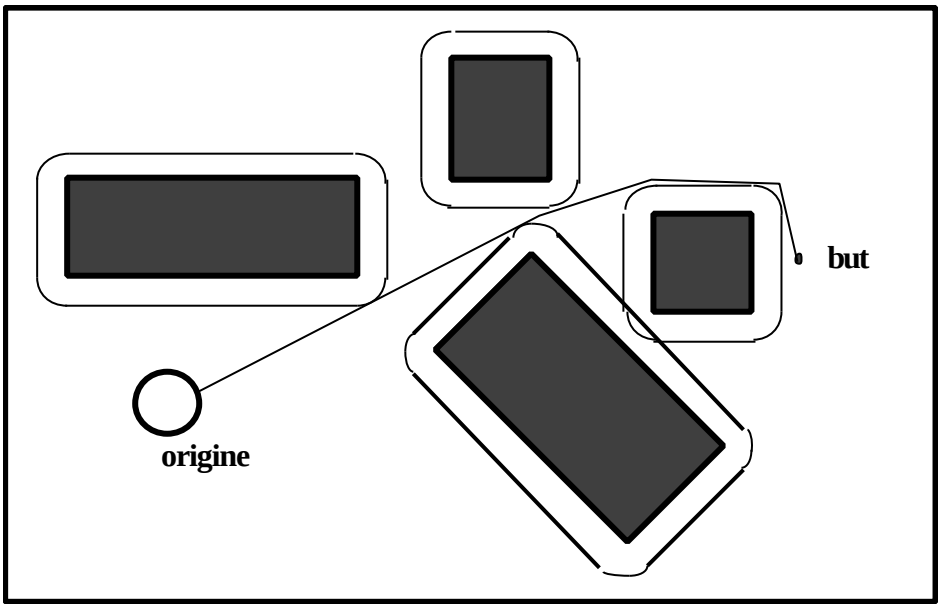


Figure 4-32. Obstacles grossis

Prenons le cas de la Figure 4-33, on voit qu'aucun chemin n'est possible entre les obstacles.

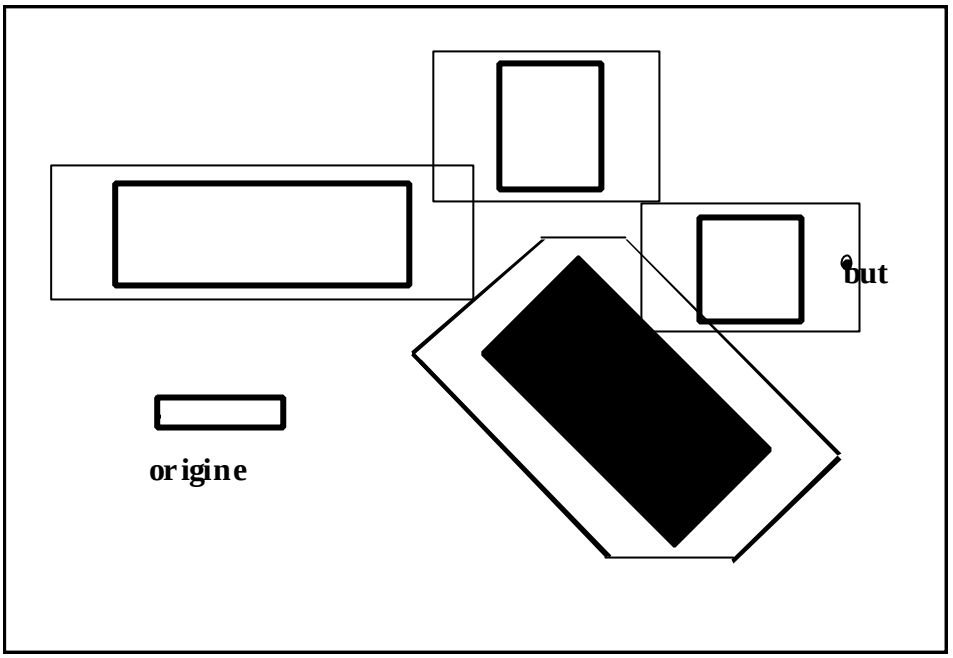


Figure 4-33. Impossible de passer

Pourtant, comme le montre la Figure 4-34, un chemin est possible quand l'objet peut tourner.

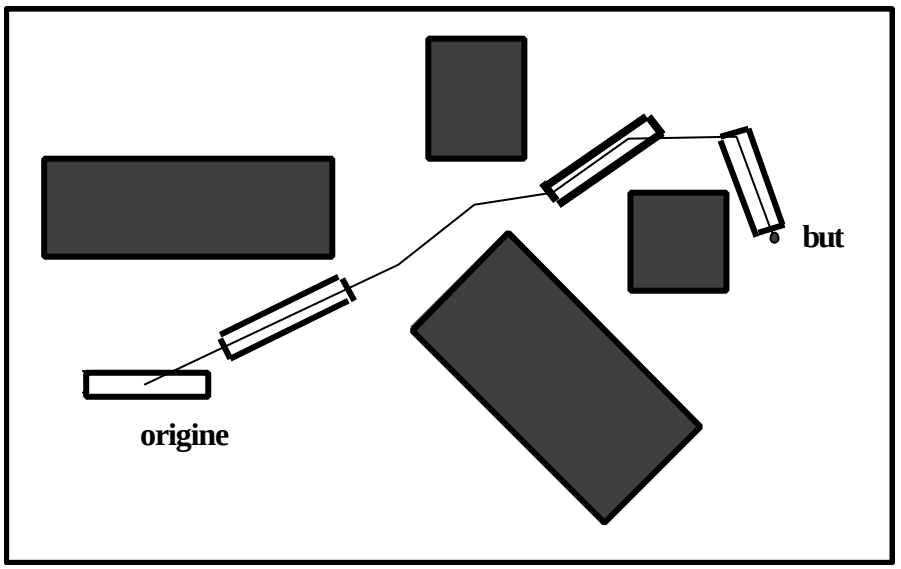


Figure 4-34. Avec rotations

La Figure 4-35 nous montre un exemple complet résolu à l'aide d'un programme implantant l'algorithme de Lozano-Perez avec obstacles grossis et rotations.

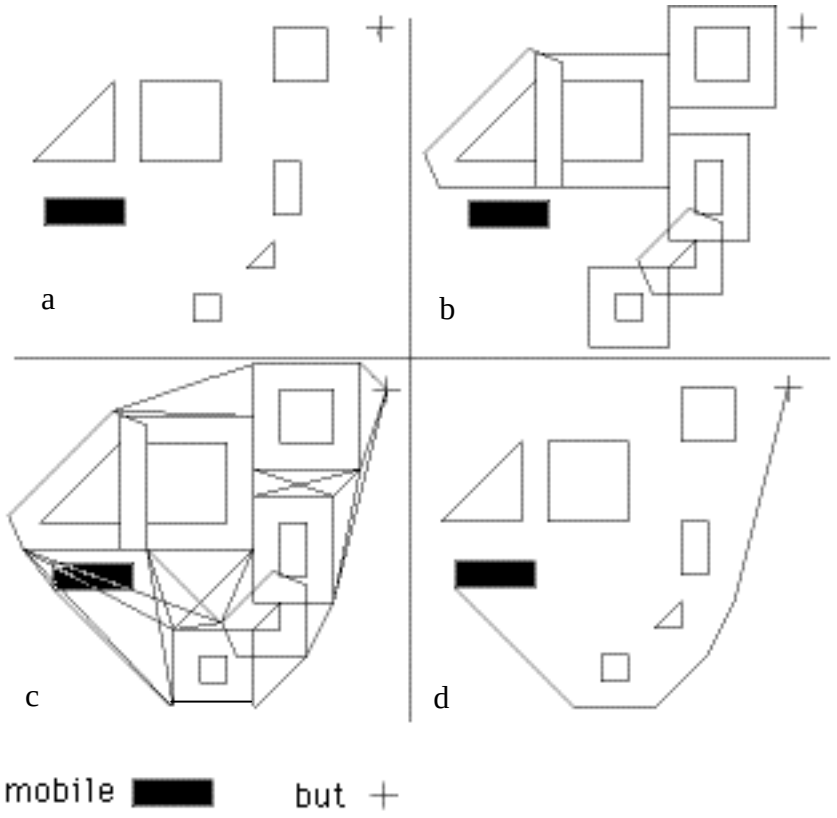


Figure 4-35. Un exemple complet résolu par ordinateur: a. Obstacles, mobile et but. B. Obstacles grossis. C. Graphe de visibilité. D. Chemin final.

4.8 L’animation de niveau tâche

4.8.1 Analogie avec les systèmes robotiques

La spécification de l'animation sous forme de tâches est importante; car, c'est le seul moyen de contrôler des mouvements sans entrer dans de nombreux détails. Il faut diriger l'animation en termes de buts à atteindre plutôt que de mouvements détaillés. Comme exemples de tâches pour des acteurs virtuels, on peut citer:

- marche d’un point à un autre
- préhension d’un objet à un endroit et déplacement à un autre endroit
- dire une phrase ou avoir une émotion

Ce type de contrôle d'animation est beaucoup plus proche de notre mode de pensée, mais très éloigné de la production concrète de l'animation qui reste une variation d'angles au cours du temps. Il faut donc un passage de la tâche de haut niveau à cette variation d'angles: c'est la planification de tâches, qui est l'objet de nombreuses études dans le domaine des bras de robots. Les problèmes sont cependant différents en animation, car le nombre d'articulations traitées est beaucoup plus grand

et les corps sont déformables. Par contre, la physique du bras de robot joue un rôle primordial qui est absent en animation.

Par analogie avec les systèmes robotiques [39], on divise le planificateur de tâches en trois parties: la modélisation des acteurs et de leur environnement, la spécification des tâches et la génération de code.

La **phase de modélisation** consiste principalement à décrire la géométrie et les caractéristiques physiques des acteurs synthétiques et des objets intervenant dans la scène, comme vu auparavant. Les mouvements des acteurs qui sont possibles dépendent des contraintes dues à la présence d'objets ou d'autres acteurs. La forme des contraintes est elle-même dépendante de la forme des objets et des acteurs, ce qui nécessite une description géométrique de tous les éléments. Nous devons également prendre en considération dans la base de données des contraintes géométriques (par exemple les limites de certaines articulations). Enfin, les propriétés physiques doivent aussi entrer en considération. Par exemple, pour générer une animation réaliste correspondant à la tâche de soulever un objet, il est indispensable de connaître au moins la masse de l'objet et si possible la force de l'acteur soulevant l'objet.

Une approche programmation-objet semble la plus adaptée à un telle démarche. On peut utiliser un modèle basé sur le concept d'**attributs**. Dans un tel modèle, la scène a ses attributs, les objets et les acteurs ont leur propres attributs et l'animation est considérée comme une relation entre les attributs.

Pour **spécifier les tâches**, il faut créer un langage de commandes. La tâche de conception d'un tel langage n'est pas aisée. En effet, s'il est facile de spécifier des tâches très générales comme marcher d'un point à un autre ou saisir un objet, il est beaucoup plus difficile de décrire la manière de le faire. Par exemple, on peut marcher à grand pas, on peut saisir une bouteille par le corps ou le goulot. Cette spécification devenant trop pénible sous la forme unique de commandes; il est préférable de la compléter par un interface graphique. Par exemple, on indiquera la partie de l'objet que l'on saisit en désignant graphiquement cette région.

La **génération de code** est la phase correspondant à la production de séquences d'animation à partir de la spécification des tâches. Le passage des spécifications de haut niveau à des séquences de mouvements élémentaires se rapproche en quelque sorte des problèmes de compilation. Trois cas sont possibles: la traduction dans un code de bas niveau, la traduction dans un langage de haut niveau et l'interprétation. La première solution correspond à une production de séquences de positions-clés et à l'utilisation d'un système d'animation paramétrique. Le mécanisme de traduction est analogue aux méthodes descendantes de la compilation. Chaque tâche est décrite comme une composition de sous-tâches séquentielles, alternatives et répétitives. Ces sous-tâches sont l'analogue des symboles non-terminaux d'une grammaire. Chaque sous-tâche est elle-même une composition de sous-tâches plus simples. Certaines sous-tâches élémentaires ne sont plus décomposables et correspondent à des symboles terminaux. En fait ces sous-tâches élémentaires doivent se réduire à des séquences d'angles-clés.

Comme dans les systèmes robotiques de niveau tâche, les actions ne sont spécifiées que par leurs effets sur les objets. Les commandes au niveau tâches sont transformées en instructions de plus bas niveau comme un script pour de l'animation procédurale ou des valeurs-clés pour une approche paramétrique.

4.8.2 La planification de tâches

Le coeur d'un système d'animation de niveau tâche est le planificateur de tâches.

La planification de tâches est un problème majeur en robotique et en I.A.. La complexité du problème est directement liée à la généralité du monde virtuel de l'acteur.

Soit une description de tâches, le problème est comment décomposer la tâche en séquence de mouvements élémentaires.

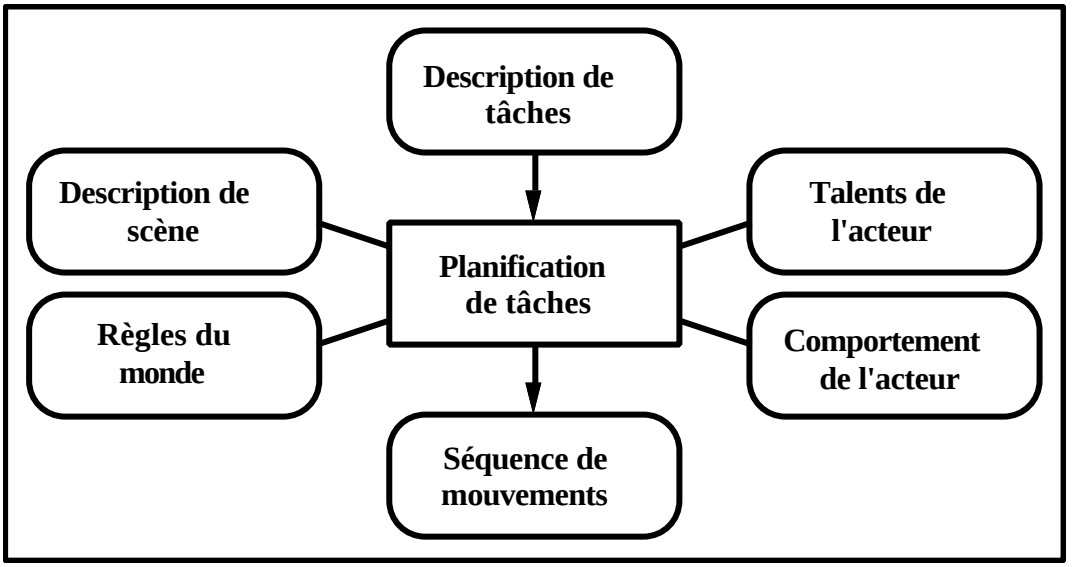


Figure 4-36. La planification de tâches

Comme le montre la Figure 4-36, pour générer ces mouvements, le système doit posséder les informations suivantes:

- la description de la scène (topologie, position et orientation des objets)
- la base de données des règles gouvernant le monde virtuel (p.e. il est nécessaire d'être debout pour marcher)
- le comportement de l'acteur (devrait influer la manière d'effectuer les mouvements; correspond à des paramètres de style)
- la bibliothèque de mouvements élémentaires pouvant être effectué par l'acteur (talents de l'acteur)

Exemple: tâche "répondre au téléphone"

Elle peut être décomposée en la séquence suivante d'actions élémentaires:

- se lever de sa chaise
- déterminer une trajectoire qui évite les obstacles lors du mouvement de l'acteur
- marcher selon cette trajectoire
- déterminer une trajectoire qui évite les obstacles dans la préhension du récepteur téléphonique
- saisir le récepteur

- répondre

Pour expliquer le mécanisme de décomposition de tâches en actions élémentaires, on prend l'exemple des blocs de Winograd.

4.8.3 Les blocs de Winograd

Cet exemple A été créé par Winograd en 1972 pour comprendre les langages naturels. Il est intéressant pour l'animation humaine, car il correspond à la planification de séquences de mouvements. La Figure 4-37 nous montre la situation initiale avec la table et les blocs.

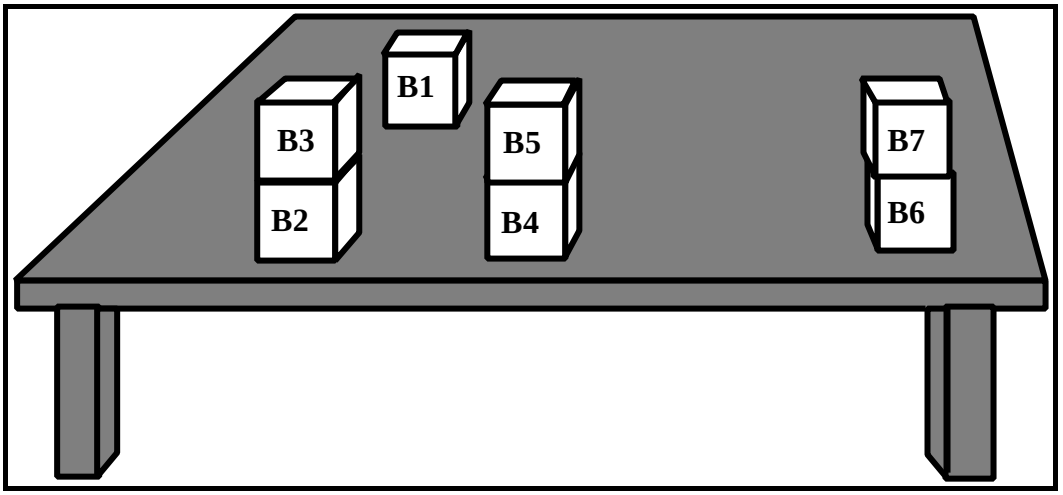


Figure 4-37. Une table et des blocs

La première tâche que nous voulons planifier est de mettre le bloc B1 sur le bloc B2. La spécification de tâche est donc PUT B1 ON B2.

On a comme commandes de plus bas niveau:

GRASP B	saisit le bloc B
MOVE B TO P	place le bloc B à la position 3D P
UNGRASP B	lâcher le bloc B

La séquence à générer est:

```
GRASP B3
MOVE B3 TO P1   (* P1 est la position 3D sur la table *)
UNGRASP B3
GRASP B1
MOVE B1 TO P2   (* P2 est la position 3D sur le bloc B2 *)
UNGRASP B1
```

On va utiliser maintenant la notation de la théorie de la compilation, les diagrammes syntaxiques. On aura, comme symboles terminaux, les commandes générées: GRASP B, MOVE B TO P, UNGRASP B.

On a un quatrième symbole terminal:

FIND SPACE FOR B ON B2 ou FIND SPACE FOR B ON TABLE

Le rôle est de trouver de la place pour le bloc B soit sur un autre bloc soit sur la table, s'il n'y a pas de place, on abandonne.

On introduit les Métacommandes (symboles non-terminaux) de la Figure 4-38.

get space on B2 for B1	trouve de la place pour B1 sur B2
put B1 on block B2 at P	place un bloc B1 à la position 3D P sur le bloc B2
put B1 on table at P	place un bloc B1 à la position 3D P sur la table
make space for B1 on B2	fait de l'espace pour B1 sur B2
get rid of B	trouve de la place pour B sur la table et y place B
clear top of B	nettoie le sommet du bloc B

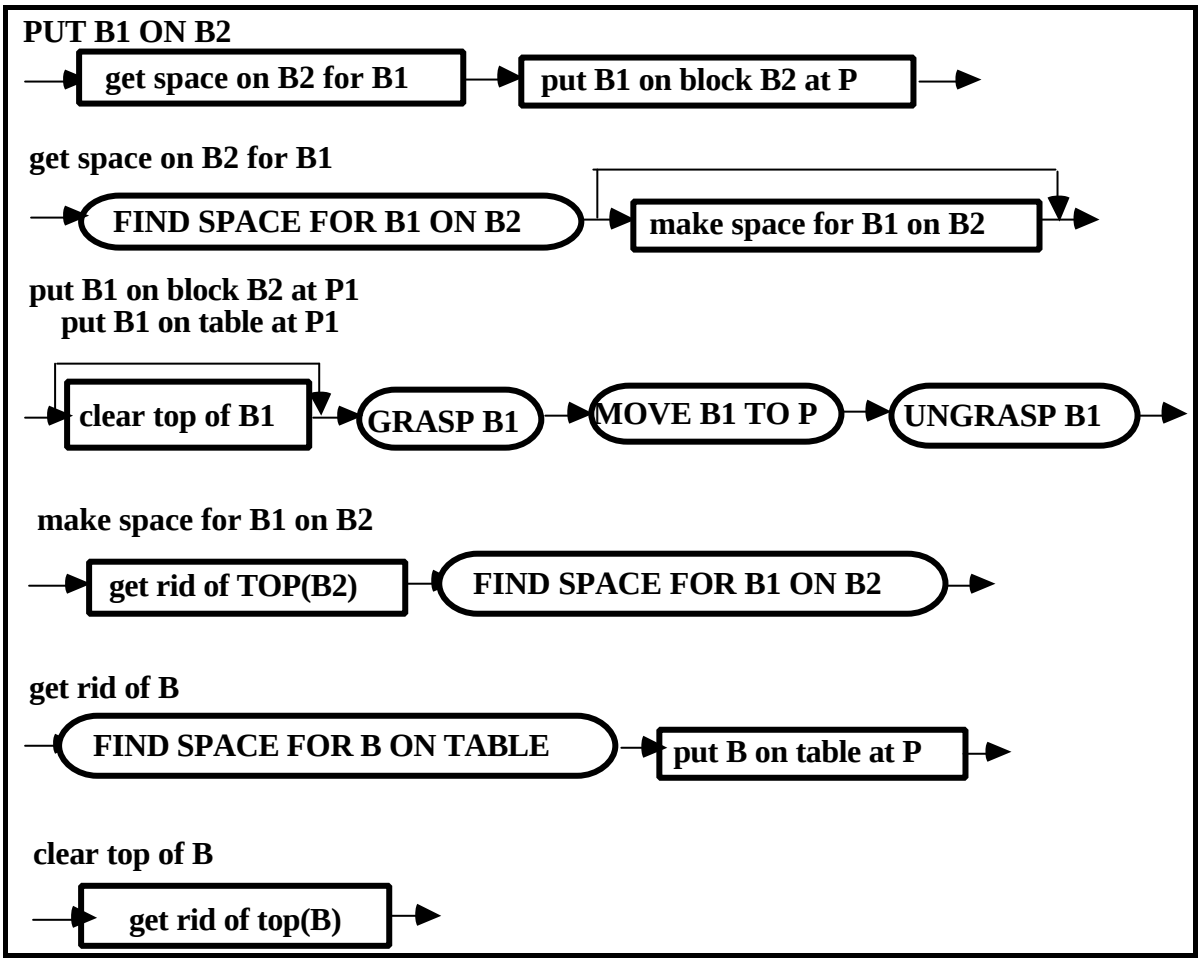


Figure 4-38. La grammaire

Si on suit une descente récursive, comme dans la construction de compilateurs, on obtient un arbre pour la tâche PUT B1 on B2 comme le montre la Figure 4-39.

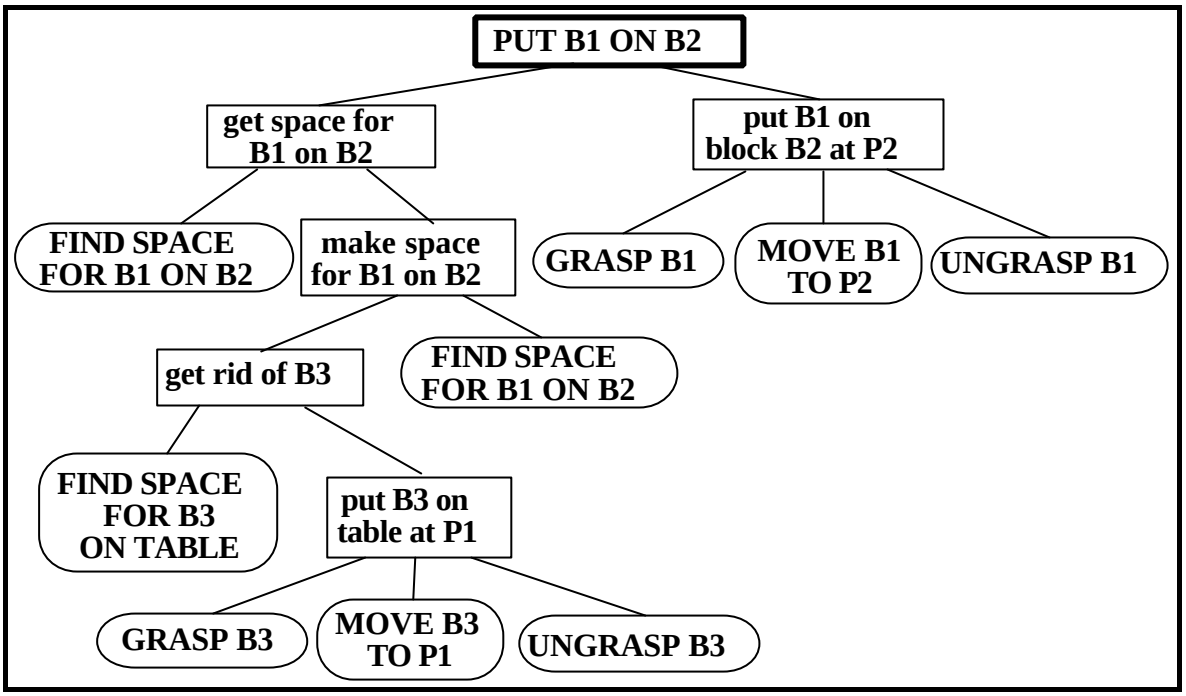


Figure 4-39. Un exemple d'arbre par descente récursive

De même, pour la commande PUT B4 ON B6, on a la séquence générée:

```
GRASP B5
MOVE B5 TO P1 (*P1 est une position 3D sur la table*)
UNGRASP B5
GRASP B7
MOVE B7 TO P2 (*P2 est une position 3D sur la table*)
UNGRASP B7
GRASP B4
MOVE B4 TO P3 (*P3 est une position 3D sur le  bloc B6*)
UNGRASP B4
```

4.8.4 La conception de langages de spécification de tâches

Les commandes typiques d'un tel langage sont:

walk to <emplacement>
put <objet> on <objet>
pick up <objet>
sit down
stand up
say <phrase>

Essayons d'appliquer la commande "put GLASS on TABLE" à l'actrice de synthèse Marilyn:

Deux cas sont possibles (exemples):

1. Marilyn est près de la table, elle a un verre dans sa main, elle a juste à effectuer la tâche.

2. Marilyn est assise sur une chaise et le verre est sur le bar loin d'elle, une séquence complexe d'actions a effectuée:

se lever, marcher jusqu'au bar, saisir le verre, marcher jusqu'à la table, poser le verre sur la table

La séquence n'est pas difficile à générer, mais elle requiert une base de connaissances consistant de la description de l'environnement.

P.e. ABOVE(GLASS, BAR)

SIT_DOWN (MARILYN, CHAIR)

Cela requiert les positions de l'actrice et des objets

Un problème est l'incertitude: p.e. où placer exactement le verre sur la table? Quel est le sens exact de "walk to TABLE"?

4.8.5 L'utilisation de relations spatiales symboliques

Les tâches sont définies comme des séquences d'états du monde. Chaque état est donné par la configuration de tous les objets de l'environnement

Il y a des méthodes pour obtenir les contraintes de configuration à partir de relations spatiales symboliques:

Considérons par exemple deux blocs B1 et B2 (Figure 4-40).

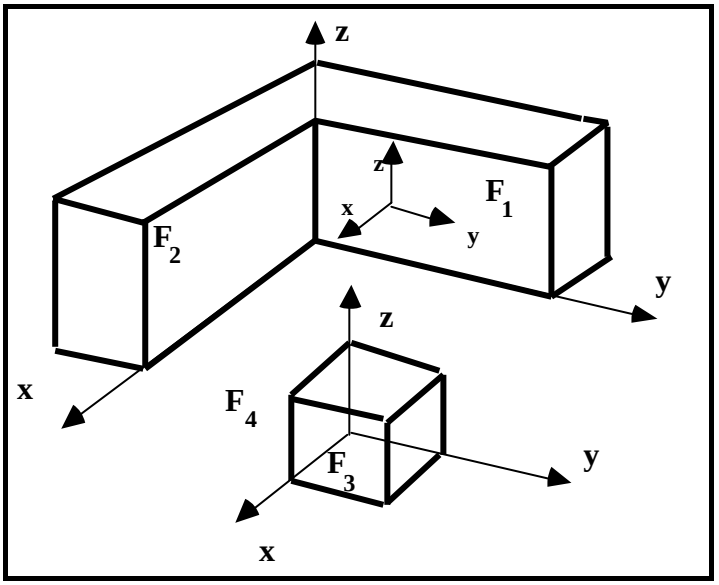


Figure 4-40. Blocs à assembler

Les configurations légales doivent satisfaire:

$$(F_3 \text{ against } F_1) \text{ and } (F_4 \text{ against } F_2)$$

Une commande de niveau tâche peut être:

place B1 so (F_3 against F_1) and (F_4 against F_2)

On peut déterminer et résoudre les équations (contraintes indépendantes sur la configuration de B1) qui doivent être satisfaites simultanément.

La relation s'exprime naturellement, car chaque bloc a des faces naturelles. Mais comment peut-on exprimer l'action exacte de saisir un objet, quand la main et l'objet sont modélisée par un grand nombre de facettes ?

Par exemple, idéalement, on devrait avoir:

saisir la BOUTEILLE de telle manière que les doigts touchent le GOULOT

ou plus simplement:

saisir la BOUTEILLE par le GOULOT

4.9 Animation comportementale

4.9.1 Introduction

Même si un mouvement est spécifié sous forme de tâche et réalisé au moyen de programmes moteurs reposant sur la cinématique et la dynamique, il n'en reste pas moins de nature très mécanique et par conséquent correspond mieux à la simulation de robots qu'à la simulation d'êtres vivants. Il est donc essentiel d'apporter une contribution comportementale dans l'animation. L'animation comportementale correspond à modéliser le comportement des personnages ou animaux (vie artificielle). Elle va de la planification de chemins à des interactions émotionnelles complexes entre personnages. L'animateur est responsable pour la conception des comportements. Son travail correspond à celui d'un metteur en scène. La performance de l'acteur est le résultat indirect des instructions du metteur en scène à l'acteur. Selon la personnalité de l'acteur, ses réactions peuvent éventuellement causer parfois des surprises.

A un niveau comportemental, nous devrons prendre en compte le comportement de groupe au même titre que le comportement individuel. Par exemple, dans la tâche de marcher, chacun marche plus ou moins de la même façon, suivant plus ou moins les mêmes lois. C'est ce "plus ou moins" qui est difficile à modéliser. De plus, une personne ne marche pas de la même façon chaque jour. Si une personne est fatiguée ou joyeuse, ou vient d'apprendre une bonne ou une mauvaise nouvelle, la manière de marcher change un peu. Ainsi dans le futur, un nouveau défi est ouvert dans l'animation des personnages: modéliser le comportement humain en prenant en considération les différences sociales et les individualités.

Dans une implantation idéale d'animation comportementale, il est impossible de jouer la même scène deux fois exactement la même chose. Vous ne pouvez pas marcher exactement la même chose quand vous revenez du même bar à la maison pour la seconde fois.

4.9.2 Modèle distribué comportemental de Reynolds (1987)

Reynolds [40] simule les nuées d'oiseaux, les troupeaux d'animaux et les bancs de poissons. Un groupe est l'élaboration d'un système de particules avec acteurs (oiseaux ou poissons) en tant que particules. Un groupe est supposé le résultat de l'interaction entre les comportements des acteurs individuels. De façon individuelle, les acteurs tentent de rester ensemble et d'éviter les collisions avec les autres et les objets aux alentours. Les positions, vitesses et orientations des acteurs sont connues du système à tout instant. L'animateur peut contrôler plusieurs paramètres globaux:

- le poids de la composante évitement d'obstacles
- le poids de la convergence vers le but
- le poids du centrage du groupe
- le poids de l'égalité des vitesses
- la vitesse maximale
- l'accélération maximale
- la distance minimale entre les oiseaux.

L'animateur fournit des données sur la trajectoire d'un "leader" et le comportement des autres oiseaux par rapport au leader.

Plusieurs films ont été produits avec une telle technologie:

Breaking the ice (Symbolics, Whitney-Demos) et *Eurhytmy* (Girard-Amkraut)

La Figure 4-41 nous montre un exemple d'animation comportementale.

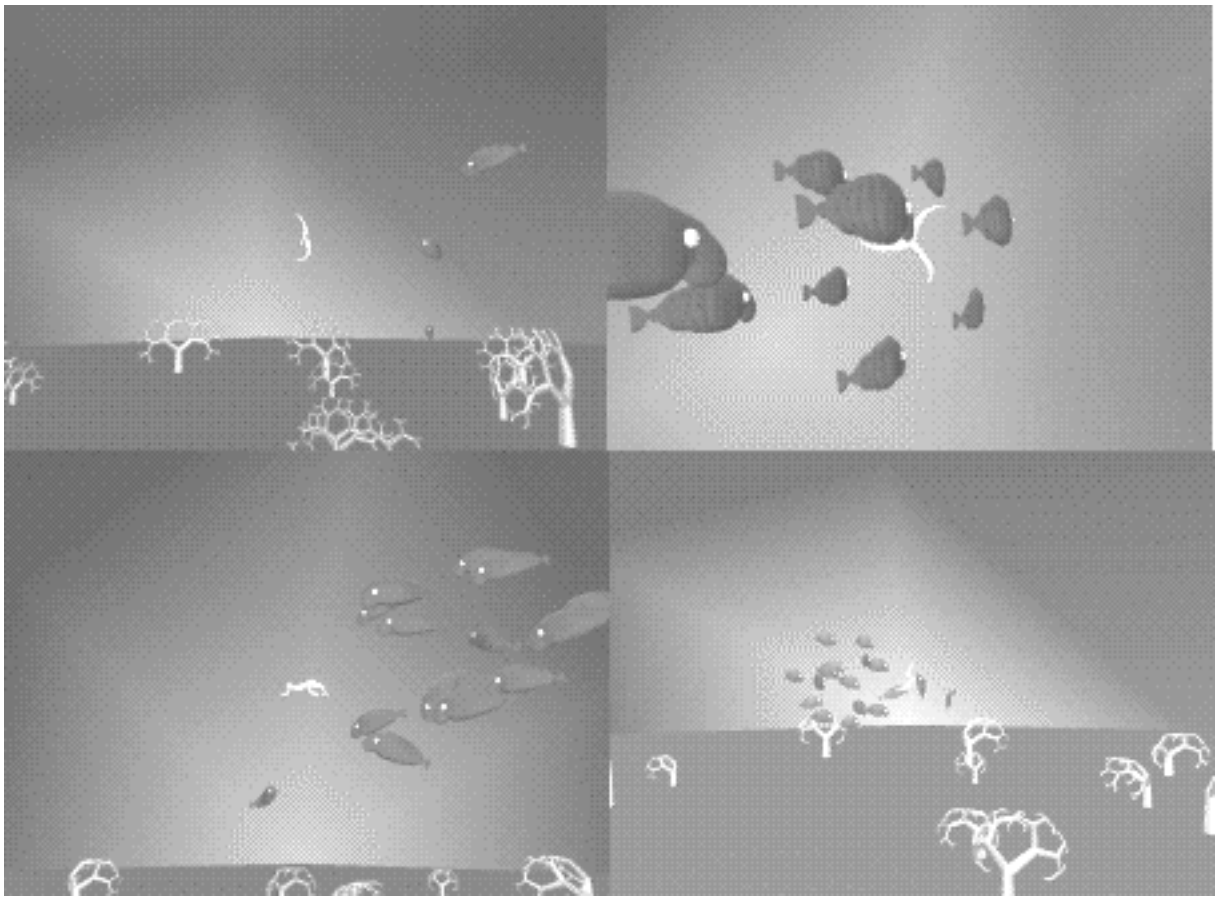


Figure 4-41. Animation comportementale de poissons

4.9.3 Animation comportementale pour des humains virtuels

Les humains virtuels ne sont pas seulement visuels mais, ce sont des acteurs intelligents et autonomes qui réagissent à l'environnement et prennent des décisions basées sur leur système perceptif, leur mémoire et leur raisonnement.

Intelligence: l'acteur est capable de planifier et d'exécuter des tâches basées sur le modèle de l'état courant du monde virtuel.

Autonomie: l'acteur ne demande pas une intervention continuelle de l'animateur.

Le but spécifique est de permettre à l'acteur d'explorer des environnements inconnus et de construire des modèles mentaux et des cartes et plans à partir de cette exploration.

Une fois les cartes et plans construits, l'acteur peut facilement se déplacer, trouver des choses ou jouer à des jeux stratégiques.

Le comportement humain typique peut se baser sur trois composantes:

- *Le système locomoteur* concerné par comment on anime les mouvements physiques de l'humain dans son environnement.

- *Le système perceptuel* concerné par la perception de l'environnement avec les modes d'attention: s'orienter, entendre, toucher, sentir, goûter, et regarder.
- *Le système organique* concerné par les règles, les talents, les motivations, la mémoire => c'est en quelque sorte le cerveau de l'acteur.

4.9.4 La boucle d'animation comportementale

Une simulation est produite en mode synchrone par la boucle comportementale suivante:

```
t_global = 0.0

code pour initialiser l'environnement de l'animation

tantque (t_global < t_final) {

code pour mettre à jour la scène

      pour chaque acteur

            code pour réaliser la perception de l'environnement

            code pour sélectionner les actions sur la base des
            entrées sensorielles, de l'état courant et du
            comportement spécifique

      pour chaque acteur

         code exécutant les actions sélectionnées au-dessus

      t_global += t_interval

}
```

Le temps global t_global sert de paramètre de synchronisation pour les différentes actions et événements. Chaque itération représente un petit pas de temps. Les actions à exécuter par chaque acteur son sélectionnées par le modèle comportemental à chaque pas de temps.

La sélection des actions s'opère en trois phases.

1. l'acteur perçoit les objets et les autres acteurs dans l'environnement, ce qui fournit de l'information sur la nature et la position des objets.

2. l'information est utilisée par le modèle comportemental pour décider de l'action à exécuter, ce qui va résulter en paramètres pour une procédure de mouvement: p.e. saisir un objet ou marcher avec de nouvelles vitesse et direction.

3. Finalement, l'acteur effectue le mouvement résultant.

4.10 Animation du visage

L'utilisation de visages humains en infographie remonte à 1971, date à laquelle Chernoff a introduit une représentation de données à N dimensions par des visages en 2 dimensions. Mais c'est à Frederic Parke du New York Institute of Technology (NYIT) que l'on doit les premières recherches sur l'animation de visages en 3 dimensions. Dans une première tentative, Parke [41] a produit une séquence d'animation en collectant des données d'expression faciales par des techniques photogrammétriques, puis en les interpolant linéairement point à point. Il a ensuite proposé le premier modèle paramétrique d'animation faciale permettant ainsi d'animer des visages différents avec toute une série d'expressions variées. L'approche paramétrique a ses limites, car elle repose strictement sur un modèle géométrique sans réalité ni physique, ni physiologique. D'autres chercheurs se sont donc dirigés vers une autre approche basée sur les muscles. Ainsi, Platt et Badler [42] ont proposé le premier modèle d'expression faciale contrôlée par des muscles. Ces muscles sont très simples et agissent comme de simples ressorts; ils fonctionnent assez bien pour la partie supérieure du visage où la nature élastique des muscles et de la peau est primordiale. Par contre, l'approche n'est pas concluante pour la partie inférieure du visage et ne peut modéliser correctement les joues et surtout la mâchoire. Waters [43] a introduit un modèle plus raffiné de muscles simulant au moyen de quelques paramètres dynamiques les principaux types de muscles. Chacun des muscles résulte alors en déplacements de la surface.

Ces recherches sont dans le domaine des déformations de base du visage; à un niveau plus élevé, il faut considérer les expressions faciales proprement dites qu'on peut séparer en deux types: les expressions reliées à la parole (phonèmes) et les autres expressions (émotions, sourire, pleurs etc...). Pour la parole, ce sont surtout les problèmes de synchronisation des expressions avec la parole qui ont été étudiés, notamment par Lewis et Parke [44] et Hill et al. [45]. Dans le domaine des autres expressions, la recherche a été peu développée. Du point de vue psychologique, seuls les travaux de Ekman et Friesen [46] font véritablement autorité. Ils ont en effet proposé un système de codage des actions faciales pour la communication non verbale. Ainsi, les expressions de surprise, de joie ou de tristesse ont été décrites par Ekman et Friesen en termes d'actions musculaires de base. Malheureusement, la théorie repose sur des expressions statiques et ne donnent aucune information sur les aspects temporels, indispensables en animation.

Dans le domaine de l'animation faciale, Magnenat Thalmann et al. [47] ont introduit une approche basée sur des **muscles abstraits** et utilisée pour la production du film *Rendez-vous à Montréal*. Cette approche est sensiblement différente des approches de Platt-Badler et Waters, car à la place de simuler des muscles en termes de structures de données, nous avons introduit une méthode procédurale. A chaque muscle est associée une procédure spécialisée qui simule l'effet du muscle plutôt que le muscle lui-même. L'avantage de cette approche est une plus grande individualisation des muscles. Chaque muscle est traité séparément, tandis que dans les autres approches, les muscles sont regroupés en catégories. A partir de ce niveau musculaire, un second niveau est introduit, le niveau des expressions, séparé en expressions de la parole et en émotions. Par exemple, pour spécifier l'expression d'un visage prononçant le son 'M", il faut donner une valeur de paramètres correspondant à une mâchoire légèrement ouverte et les lèvres fermées; on sélectionne donc 15% d'ouverture de la mâchoire, 100% de fermeture de la lèvre supérieure et 100% de fermeture de la lèvre inférieure. Le troisième niveau correspond à la définition temporelle de l'animation, c'est-à-dire à la spécification des activations et des durées des expressions. Par exemple, pour prononcer "MI", il faut que le phonème "M" soit sélectionné à un temps donné, puis le phonème "I" un instant plus tard. Le logiciel déforme par interpolation de spline l'expression du visage du phonème "M" pour qu'il se transforme progressivement en l'expression du phonème "I".

On aura aussi des expressions correspondant a différentes émotions et état d'âme: sourire, angoisse, peur, joie, tristesse, colère (Figure 4-42).

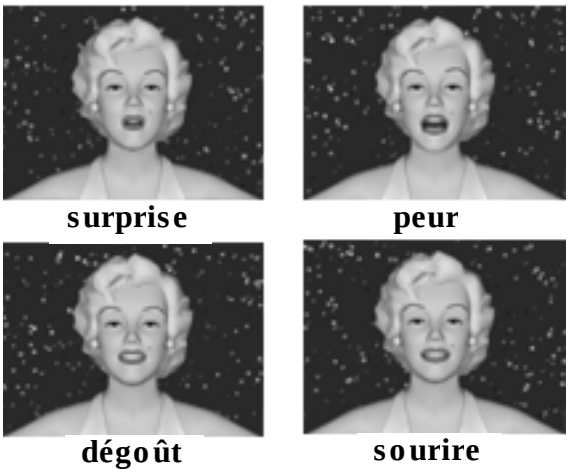


Figure 4-42. Facial expressions

4.11 References on animation

1. Baecker R (1969) Picture-driven Animation, Proc. AFIPS Spring Joint Comp. Conf., Vol.34, pp.273-288

2. Burtnyk N, Wein M (1976) Interactive Skeleton Techniques for Enhancing Motion Dynamics in Key Frame Animation, Comm. ACM, Vol.19, No10, pp.564-569.

3. Reeves WT (1981) Inbetweening for computer animation utilizing moving point constraints. Proc. SIGGRAPH '81, Computer Graphics, Vol.15, No3, pp.263-269

4. Steketee SN, Badler NI (1985) Parametric Keyframe Interpolation Incorporating Kinetic Adjustment and Phrasing Control, Proc. SIGGRAPH '85, pp. 255-262.

5. Wolberg G (1990) Digital Image Warping, IEEE Computer Society Press.

6. Kochanek D and Bartels R (1984) Interpolating Splines with Local Tension, Continuity and Bias Tension, Proc. SIGGRAPH '84, Computer Graphics, Vol.18, No3, pp.33-41.

7. Reynolds CW (1982) Computer Animation with Scripts and Actors, Proc. SIGGRAPH'82, Computer Graphics, Vol.16, No3, pp.289-296.

8. Magnenat-Thalmann N, Thalmann D (1983) The Use of High Level Graphical Types in the MIRA Animation System, IEEE Computer Graphics and Applications, Vol. 3, No 9, pp. 9-16.

9. Magnenat-Thalmann N, Thalmann D, Fortin M (1985) MIRANIM: An Extensible Director-Oriented System for the Animation of Realistic Images, IEEE Computer Graphics and Applications, Vol.5, No 3, pp. 61-73.

10. Magnenat-Thalmann N, Thalmann D (1985) 3D Computer Animation: More an Evolution Problem than a Motion Problem, IEEE Computer Graphics and Applications Vol. 5, No10, pp. 47-57

11. Magnenat-Thalmann N, Thalmann D (1986) Special cinematographic effects using multiple virtual movie cameras, IEEE Computer Graphics and Applications 6(4):43-50

12. Turner R, Balaguer F, Gobbetti E, Thalmann D (1991), Physically-Based Interactive Camera Motion Control Using 3-D Input Devices, in: Patrikalakis N (ed.) Scientific Visualization of Physical Phenomena, Springer, Tokyo, pp.135-145.

13. Magnenat Thalmann N, Thalmann D (1991) Complex Models for Animating Synthetic Actors, IEEE Computer Graphics and Applications, Vol.11, No5, pp.32-44.

14. Magnenat-Thalmann N, Thalmann D (1987) The Direction of Synthetic Actors in the Film Rendez-vous à Montréal, IEEE Computer Graphics and Applications, Vol.7, No12, pp.9-19

15. Chadwick J, Haumann DR, Parent RE (1989) Layered Construction for Deformable Animated Characters, Proc. SIGGRAPH '89, Computer Graphics, Vol. 23, No3, pp.234-243

16. Shen J, Thalmann D (1995) Interactive Shape Design Using Metaballs and Splines, Proc. Eurographics Workshop on Implicit Surfaces '95 , Grenoble, pp.187-196.

17. Featherstone R (1983) Position and Velocity Transformations Between Robot End-Effector Coordinates and Joint Angles, Intern. Journal of Robotics Research, Vol.2, No2, pp.35-45.

18. Boisvert D, Magnenat-Thalmann N, Thalmann D (1989) An Integrated View of Synthetic Actors, in: Earnshaw E, Wyvill B, New Advances in Computer Graphics, Springer, Tokyo, pp.277-288

19. Badler NI et al. (1986) Multi-Dimensional Input Techniques and Articulated Figure Positioning by Multiple Constraints, Workshop on Interactive 3D Graphics, Chapel Hill, North Carolina

20. Thalmann D (1991) Dynamic Simulation as a Tool for Three-Dimensional Animation, in: Magnenat Thalmann N, Thalmann D (eds) New Trends in Animation and Visualization, John Wiley, Chichester, UK, pp.257-271.

21. Thalmann D (1990) Robotics Methods for Task-level and Behavioral Animation, in: Thalmann D (ed) Scientific Visualization and Graphics Simulation, John Wiley, Chichester, UK, pp.129-147.

22. Stepanenko Y and Vukobratovic M (1976) Dynamics of Articulated Open Chain Active Mechanisms, Mathematical Biosciences, Vol.28, pp.137-170.

23. Orin D, McGhee R, Vukobratovic M and Hartoch G (1979) Kinematic and Kinetic Analysis of Open-Chain Linkages Utilizing Newton-Euler methods, Mathematical Biosciences, Vol.31, pp.107-130.

24. Armstrong WW (1979) Recursive Solution to the Equations of Motion of an N-Link Manipulator, Proc. 5th World Congress Theory Mach. Mechanisms, Vol.2, pp.1343-1346

25. Luh JYS, Walker MW and Paul RPC (1980) On-line Computational Scheme for Mechanical Manipulators, Journal of Dynamic Systems, Measurement and Control, Vol.102, pp.103-110.

26. Uicker JJ (1965) On the Dynamic Analysis of Spatial Linkages Using 4x4 Matrices, Ph.D Dissertation, Northwestern University, Evanston, IL.

27. Kahn ME (1969) The Near-Minimum-Time Control of Open-Loop Articulated Kinematic Chains, Stanford Artificial Intelligence project, AIM-106

28. Hollerbach JM (1980) A Recursive Lagrangian Formulation of Manipulator Dynamics and a Comparative Study of Dynamics Formulation Complexity, IEEE Trans. on Systems, Man, and Cybernetics, Vol. SMC-10, No11, pp.730-736

29. Armstrong WW, Green M, Lake R (1987) Near real-time Control of Human Figure Models, IEEE Computer Graphics and Applications, Vol.7, No 6, pp.28-38

30. Wilhelms J and Barsky B (1985) Using Dynamic Analysis to Animate Articulated Bodies such as Humans and Robots, in: N.Magnenat-Thalmann and D.Thalmann (Eds) Computer-generated Images, Springer, pp.209-229.

31. Arnaldi B, Dumont G, Hégron G, Magnenat-Thalmann N, Thalmann D (1989) Animation Control with Dynamics, in: Magnenat-Thalmann N, Thalmann D (eds) State-of-the-Art in Computer Animation, Springer, Tokyo, pp.113-124.

32 Isaacs PM, Cohen MF (1988) Mixed Methods for Complex Kinematic Constraints in Dynamic Figure Animation, The Visual Computer, Vol. 4, No6, pp.296-305.

33. Witkin A, Fleischer K, Barr A (1987) Energy Constraints on Parameterized Models, Proc. SIGGRAPH'87, Computer Graphics, Vol.21, No4, pp.225-232

34. Barzel R, Barr AH (1988) A Modeling System Based on Dynamic Constraints, Proc. SIGGRAPH '88, Computer Graphics, Vol.22, No4, pp.179-188

35. Platt JC, Barr AH (1988) Constraint Method for Flexible Models, Proc. SIGGRAPH '88, Computer Graphics , Vol. 22, No4 , pp.279-288.

36. Witkin A, Kass M (1988) Spacetime Constraints, Proc. SIGGRAPH '88, Computer Graphics, Vol.22, No4 , pp.159-168.

37. Boulic R, Huang Z, Magnenat Thalmann N, Thalmann D (1994) Goal Oriented Design and Correction of Articulated Figure Motion with the TRACK system, Computers and Graphics, Pergamon Press, Vol.18, No4, pp.443-452.

38. Lozano-Perez T, Wesley MA (1979) An Algorithm for Planning Collision-Free Paths Among Polyhedral Obstacles, Comm.ACM, Vol.22, No10, pp. 560-570.

39. Lozano-Perez T (1982) Task Planning in: Brady M (Ed.) Robot Motion: Planning and Control, MIT Press, Cambridge, Mass.

40. Reynolds CW (1987) Flocks, herds, and schools: a distributed behavioral model, Proc. SIGGRAPH '87, Computer Graphics, Vol.4, pp.25-34

41. Parke FI (1975), A Model for Human Faces that allows Speech Synchronized Animation, Computer and Graphics, Pregamon Press, Vol. 1, No. 1, pp. 1-4.

42. Platt S, Badler N (1981), Animating Facial Expressions, Proc. SIGGRAPH '81, pp.245-252.

43. Waters K (1987), A Muscle Model for Animating Three Dimensional Facial Expression, Proc SIGGRAPH '87, Vol. 21, No. 4, pp. 17-24.

44. Lewis JP, Parke FI (1987), Automated Lipsynch and Speech Synthesis for Character Animation, Proc. CHI '87 and Graphics Interface '87, Toronto, pp. 143-147.

45. Hill DR, Pearce A, Wyvill B (1988), Animating Speech: An Automated Approach Using Speech Synthesised by Rules, The Visual Computer, Vol. 3, No. 5, pp. 277-289.

46. Ekman P, Friesen WV (1978) Facial Action Coding System, Investigator's Guide Part 2, Consulting Psychologists Press Inc.

47. Magnenat-Thalmann N, Primeau E, Thalmann D (1988), Abstract Muscle Action Procedures for Human Face Animation, The Visual Computer, Vol. 3, No. 5, pp. 290-297.