Direct Animation Interfaces:
An Interaction Approach to Computer Animation

Dissertation

by
Benjamin Walther-Franks

Submitted in partial fulfilment of the requirements for the
Degree of Doctor of Engineering (Dr.-Ing.)

Faculty 3: Mathematics/Computer Science
Universität Bremen

1st reviewer: Prof. Dr. Rainer Malaka, Universität Bremen
2nd reviewer: Prof. Dr. Jan Borchers, RWTH Aachen

Date of doctoral viva: 31st July 2014

This work was funded in part by the Klaus Tschira Stiftung.

Abstract

Creativity tools for digital media have been largely democratised, offering a range from beginner to expert tools. Yet computer animation, the art of instilling life into believable characters and fantastic worlds, is still a highly sophisticated process restricted to the spheres of expert users. This is largely due to the methods employed: in keyframe animation dynamics are indirectly specified over abstract descriptions, while performance animation suffers from inflexibility due to a high technological overhead. The reverse trend in human-computer interaction to make interfaces more direct, intuitive, and natural to use has so far hardly touched the animation world: decades of interaction research have scarcely been linked to research and development of animation techniques.

The hypothesis of this work is that an interaction approach to computer animation can inform the design and development of novel animation techniques. Three goals are formulated to illustrate the validity of this thesis. Computer animation methods and interfaces must be embedded in an interaction context. The insights this brings for designing next generation animation tools must be examined and formalised. The practical consequences for the development of motion creation and editing tools must be demonstrated with prototypes that are more direct, efficient, easy-to-learn, and flexible to use.

The foundation of the procedure is a conceptual framework in the form of a comprehensive discussion of the state of the art, a design space of interfaces for time-based visual media, and a taxonomy for mappings between user and medium space-time. Based on this, an interaction-centred analysis of computer animation culminates in the concept of direct animation interfaces and guidelines for their design. These guidelines are tested in two point designs for direct input devices. The design, implementation and test of a surface-based performance animation tool takes a system approach, addressing interaction design issues as well as challenges in extending current software architectures to support novel forms of animation control. The second, a performance timing technique, shows how concepts from video browsing can be applied to motion editing for more direct and efficient animation timing.

Zusammenfassung

Für die meisten Medientypen steht eine große Bandbreite von digitalen Gestaltungswerkzeugen zur Verfügung, von einfachen Tools für Gelegenheitsnutzer bis zu leistungsfähiger Software für den professionellen Einsatz. Die Computeranimation, als die Kunst der animierten Darstellungen glaubwürdiger Figuren und fantastischer Welten, bleibt fast ausschließlich Spezialisten vorbehalten. Wichtige Gründe dafür liegen in den Animationsverfahren, die eine indirekte Steuerung der Dynamik über abstrahierte Beschreibungen erfordern (Keyframe-Animation), beziehungsweise mit hohem technischen Aufwand verbunden und daher wenig flexibel sind (Performance Animation). Der gegenläufige Trend in der Mensch-Technik-Interaktion, den Gebrauch von Benutzerschnittstellen direkter, intuitiver und natürlicher zu machen hat die Forschung im Bereich der Computeranimation bisher kaum beeinflusst.

Der vorliegenden Arbeit liegt die Hypothese zugrunde, dass Animationswerkzeuge der nächsten Generation vom Nutzer und der Interaktion her gedacht werden müssen. Zur Beschreibung diese Ansatzes werden drei Ziele formuliert: Verfahren und Werkzeuge der Computeranimation sollen in einen Interaktionskontext eingebettet werden. Die Erkenntnisse, die diese Perspektive bringt sollen erforscht, zusammengetragen und als Gestaltungsrichtlinien formuliert werden. Anhand von Prototypen soll illustriert werden wie mittels dieser Richtlinien direktere, effizientere, einfachere und flexiblere Werkzeuge zur Bearbeitung von Animationen erstellt werden können.

Eine umfassende Darstellung des Standes der Wissenschaft und Technik, ein Gestaltungsraum von Bedienschnittstellen zur Bearbeitung zeitabhängiger visueller Medien, sowie eine Taxonomie der Abbildungen zwischen realer und medialer Raum-Zeit bilden die Grundlage des Vorgehens. Auf dieser Basis erfolgt eine interaktionszentrierte Analyse von Animationsverfahren, die zu einem Konzept von *Direct Animation Interfaces* und dazugehörigen Gestaltungsrichtlinien führt. Diese werden in zwei Beispieldesigns angewendet. In der Gestaltung, Umsetzung und dem Test eines Animationstools für interaktive Oberflächen wird neben Fragestellungen der Schnittstellengestaltung erörtert, wie man klassische Softwarearchitekturen so erweitern kann, dass sie neue Bedienkonzepte unterstützen. Die Technik *Dragimation* zeigt, wie neue Steuerungsmöglichkeiten für die Navigation von Videos in ein Verfahren für direktere und effizientere Gestaltung von Bewegungsdynamiken übertragen werden können.

Contents

List of Figures

Chapter 1

Introduction

Animation is all around us. From film and television to computer games and the world wide web, the illusion of life is in demand as it has never been before. This demand gives rise to new requirements. A century ago, an elite of experts defined how animation was practiced. Today, the possibilities, consumer expectations, and number of artists have increased immensely. It is likely this development will continue in the future, as the power of dynamic imagery moves into ever more areas of modern life.

In the early days of animation, scenes were brought to life image by image. Computerisation introduced the third dimension, and eliminated the need to laboriously define every single frame. Traditional animation has a long history, and the principles of successful frame-based animation are well understood for hand-drawn frame animation and its digital successor keyframe animation (Thomas and Johnston, 1981; Lasseter, 1987). Yet in its essence this form of animation has not changed much since the first half of the last century. Computing power and advancements in technology have brought acting and puppetry back to the art of animation. In performance animation, an actor's performance is captured and used to animate a digital character. This method is widely used in film, television, and computer games productions today (Menache, 2011). A significant part of modern animation is automated by simulations of real-world processes, such as the motion of objects, materials, or even humans. While programming is a powerful tool for creating realistic, complex animation, the defining accents of bringing the inanimate to life is still crafted and designed by artists rather than algorithms.

A trend in research on computer animation interfaces criticises the complex and abstract nature of frame-based motion design tools and the limitations of current performance animation practice. This mirrors a general trend in human-computer interaction to make interfaces more direct, intuitive, and natural to use (Van Dam, 1997; Frohlich, 1997; Naumann et al., 2007; Wigdor and Wixon, 2011). While animation potentially has a lot to gain from interaction design knowledge, these parallel movements have not been connected so far.

This thesis aims to better connect computer animation to human-computer interaction (HCI) methodology by looking at user capabilities, interaction metaphors, and techniques. The rest of this chapter identifies issues of contemporary animation user interfaces (UIs) that stand in the way of a better user experience (section 1.1). An agenda for addressing these issues is proposed (section 1.2), followed by an overview of how the following chapters implement it (section 1.3).

1.1 Interaction Issues in Computer Animation

Computer animation originates in the field of computer graphics and so the vast majority of work deals with issues of modelling, representation, and rendering. However, a significant amount of work is also committed to discussing and addressing issues of interaction. This section covers the main points, and also observes higher-level issues of the underlying research approach.

1.1.1 User Interface Issues

In computer-based keyframe and performance animation, artists control the motion and behaviour of a target scene through an interface. The design of this interface significantly influences the motion design task. Interaction issues in current practice can be observed in frame animation and performance animation.

Indirections in Frame-based Animation

The strengths of keyframe animation are precision and control. Keyframe artists can specify exactly what happens, exactly when it happens, they have *absolute control*. Also, *anything is possible*: the artists imagination is the limit, anything he can conceive and draw or model can be put onto the screen. Laws of physics can be bent and broken. Gestures and facial expressions can be exaggerated. In fact, overemphasis is a large part of the principles of traditional animation (Thomas and Johnston, 1981). This "overdone reality" of dynamic caricatures has become the trademark quality of traditional animation.

However, the parametric specification of temporal events requires an *abstract* grasp of processes as time units and functions. Many rules and mappings must be simply learnt and artists animating frame by frame can use real-world intuition to only a limited extent. *Indirection* through mediation is inherent to traditional animation, both on the tool as well as on the production level. On the tool level, specifying motion in sequences of temporally spaced discrete stages makes the transfer from artist's mind to the animation highly indirect. On the production level, highly specialised production pipelines create distance between animator and the product, since each artist is only involved in a small part of the workflow (Baecker, 1969). The blessing of absolute control is also a curse—since everything needs to be defined, the artist must define everything (PIXAR, 2010). This makes frame-based animation tedious and time-consuming. The nature of explicitly determining every aspect of animation through parametric modelling causes a relation of production time to animation time that is often in the range of thousands or more to one, making this method rather *inefficient* compared to others. It is also *hard to learn*. Traditional animation is a craft that takes years, if not decades to master (Williams, 2009). Not everyone with artistic vision and expressive powers has the level of technical thinking and abstraction required for traditional animation tools.

Inflexibility in Performance Animation

Motion capture technology can track many features of real-world motion simultaneously. This high bandwidth real time input makes performance animation quite *efficient* (Jurgensen, 2008). Performance animation is more *accessible*, since non-technical people can use innate and learned abilities of their body to drive an animation. It allows for *expressive* use by tapping the innate skills and feeling humans have for space and dynamics in a much more literal manner.

Downsides lie in the way performance animation systems are currently designed and used. In common setups, a performer's body or facial motion is captured and transferred straightforwardly to a digital counterpart. While this is sufficient for humanoid digital characters, many animation tasks require more diverse and flexible puppetry controls. Mainstream performance animation is thus still *conceptually limited* regarding control mappings. Despite significant advances in tracking technology, motion capture still involves handling and maintaining complex tracking and processing hardware. Noisy dense data needs cleaning and processing for practical handling. This lets *technology dominate* development and practice, rather than focussing on performer needs and powerful controls.

1.1.2 Issues in Theory and Application

Research on motion design interfaces is spread among several disciplines. The "mother discipline" computer graphics is preoccupied with questions of modelling, computation, and rendering, and thus interaction issues can only be treated incidentally. Yet research on animation interfaces could benefit from a better connection to human-machine interaction theory and practice.

Fragmented State of the Art

Computer animation emerged as a research area in computer graphics. It has now advanced into artificial intelligence, mixed and virtual reality, and human-computer interaction. Concepts, focus and mindsets differ between these disciplines. A user-centric approach to animation tools thus requires a reframing of the state of the art into a common perspective.

Neglected User

Even though the user is central to interactive computer-based animation, the topics human factors, usability and user experience are treated subordinate to issues of algorithms for modelling, representation and rendering. The ambition to better consider the needs of the artists in computer graphics research has arisen now and again in recent publications (e.g. Popović et al., 2003; Schumacher et al., 2012).

Parallel Trends

The last two decades in HCI research has been shaped by interactions moving away from the desktop and into the world. A parallel trend in research on computer animation interfaces criticises the complex and abstract nature of current motion design tools in favour of techniques that are more accommodating to beginners and advanced users alike. These are dispersed efforts that are far from being one movement, even though they follow the common goal of making animation interfaces easier and more supportive of experimentation.

1.2 An HCI View on Computer Animation

This thesis argues that in order to design and develop next generation tools, animation needs to be approached from an HCI perspective. This will result in new ways of thinking about interactions with continuous visual media, which in turn will improve investigation of new animation interfaces based upon a theoretical foundation. This perspective will help in discovering how motion design interfaces can be made more beginner *and* expert friendly.

1.2.1 Hypothesis

The hypothesis of this work is that an HCI approach to computer animation can aid the research on novel animation techniques. This general hypothesis cannot be proven or disproven as such. However, setting goals formulated from the identified issues can aid in reaching a better assessment of its validity. The first goal is to embed computer animation methods and interfaces in an HCI context by connecting the latest trends in animation techniques and general research on next generation interfaces. Secondly, this theoretical foundation then aids an interaction-driven analysis of animation methods and techniques, the results of which can be distilled into design knowledge. The third goal is to illustrate the application of this knowledge in the development of more direct, efficient, easy-to-learn and flexible animation tools.

1.2.2 Approach

These goals are approached in three main steps. First, a conceptual framework is established with which to discuss the subject matter. This is then used to analyse established solutions and current trends, in order to identify design guidelines for next generation animation interfaces. Finally these guidelines are applied to prototype new motion design tools.

Conceptual Framework for Designing Animation Interfaces

As a basis, the entire related work from computer graphics, human-computer interaction and entertainment computing literature is reviewed. The survey is normalised by taking a user- and interface-centric approach, rather than concentrating on algorithms and technologies. It is structured by a mixture of concepts from the graphics and interaction domains: methods, mappings and metaphors. An HCI perspective on computer animation interfaces is then constructed in the form of a design space for interfaces that deal with spatiotemporal media, in which time must be controlled in addition to space. HCI concepts are extended as necessary to adequately formulate interactions with(in) space and time.

Analysis of Motion Design Interfaces

Computer animation methods and interfaces are analysed using HCI frameworks; it is explored how motion design tools can acquire desirable qualities associated with certain interface trends. The concept of direct animation interfaces is established together with a set of design principles for improving the user experience of animation tools.

Prototype Systems and Techniques

The guidelines are applied in two point designs of direct animation interfaces. The first design is an entire system for direct motion creation and editing. By taking a system approach it is explored how direct animation can be implemented in practice. The second design focusses on a novel technique for motion timing, a central task in animation. It is presented in detail and evaluated against reference techniques.

1.3 Thesis Overview

Chapter 2 presents the state of the art in computer-aided motion design and animation by focussing on interaction aspects such as techniques, mappings and metaphors employed.

Next, a design space of motion design interfaces is constructed in chapter 3. In the process it is shown that the HCI state of the art lacks effective means of modelling the relations of user and medium space-time. A novel scheme for exactly such purposes is presented and integrated into the design space.

The following chapter 4 looks at recent trends in human-computer interaction that have been summarised as post-WIMP interfaces, and investigate qualities and frameworks that characterise them. Categories of man-machine interaction in motion design are then analysed using the design space and an analysis framework for post-WIMP interfaces from the literature. This leads to criteria for the new concept of *direct animation interfaces*, which are summarised as design guidelines.

Using these design guidelines, chapter 5 makes a case for a direct-input animation system using multi-touch interactive surfaces. The design, implementation, and test of the system is documented. The process involves tackling both challenges regarding performance and view control and software engineering challenges involved in implementing direct animation on contemporary software architectures.

Chapter 6 proposes Dragimation, a novel performance-based animation timing technique for more direct temporal editing of animations. It is inspired by recent developments in object-based video navigation and addresses several issues with current timing tools. An evaluation against reference controls shows that Dragimation objectively improves timing performance as well as being the preferred solution of both novice and advanced animators.

The contributions are summarised in chapter 7, followed by a discussion of potential target users, a discourse on potential application areas and a review of recurring trade-offs in the design of animation interfaces.

1.4 Notes on Terminology

While some sources speak of only traditional methods as *animation* (Cameron et al., 1997), this thesis uses the extended definition of the term in which every method for bringing inanimate objects to life is considered animation. Thus, automated, scripted, or example-based means of specifying the motion and behaviour of virtual characters and scenes counts as animation for our purposes. Since the main task of animation is designing spatial changes over time, the term *motion design* is used interchangeably throughout this document.

Computer-based animation is animation that was created with the aid of a computer to a significant degree, i.e. as part of the motion design process itself (as opposed to modelling, lighting, rendering, cleanup etc.).

Chapter 2

Related Work

Ever since computers were established, they have also been influencing the art of animation. The computer made the animator's life easier by interpolating character poses, and opened up new possibilities for modelling and visual rendering. Computer-based frame animation is the direct successor of traditional hand-drawn animation, and still the main method: animated feature films rely on keyframe animation to a large extent. But advances in sensing hardware and processing power have brought entirely new possibilities. Motion capture records the live performance of actors, introducing a new form of animation more akin to puppetry than traditional animation. Programmed animation enables the realistic simulation of fluids, cloth and hair to provide interesting secondary motion and create more believable worlds.

Computer animation developed as a research area in computer graphics, but is now spread amongst many disciplines, with researchers in various computer science disciplines contributing new ideas on algorithms and user interfaces. Each discipline brings its own approach, mindset, terminology and values to the discussion. However, research is still strongly influenced by its graphics background. A human factors approach is uncommon, but it can aid addressing user interface issues often identified for motion design tools (section 1.1). The review of related work presented in this chapter includes the whole range of research on motion design interfaces, from user-centric HCI publications to technology-centric computer graphics literature. It is structured using graphics and interaction concepts at different levels, with a user-centric view on mappings and metaphors, rather than the algorithms working "behind the scenes".

Since they are fundamental to most computer-aided interactive visual design processes, this chapter treats general object manipulation techniques first (section 2.1), before presenting the state of the art in frame-based (section 2.2) and performance animation (section 2.3). A treatment of non-interactive animation methods follows in section 2.4. Section 2.5 considers the relation of animation and video games, and section 2.6 concludes the chapter with a discussion of contributions and limitations of the review's perspective.

2.1 Object Manipulation

Spatial manipulation techniques are a fundamental part of many computer-aided design solutions. They can be specifically designed for a certain task, or use general rigid body deformation techniques, which are the shape-preserving operations translate, rotate and scale (Bowman et al., 2004). Interactive spatial manipulation techniques create a relation between user input and target object through a device and a software mapping. They can be grouped into kinematic techniques, that more or less directly transfer the change of spatial input device parameters to object transformation, and physics-based techniques in which user-object interaction is mediated in a simulation of the laws of physics. Kinematic techniques can be further differentiated based on the degree of coordination—integral, separated or constraint-based.

2.1.1 Integrated Control

Object manipulation techniques for desktop input devices typically do not offer the high number of degrees of freedom (DOF) required for integrated control. This can be addressed by multi-touch input devices that bring multi-DOF control into the reach of many users. Researchers have been exploring the possibilities of multi-point control regarding integrated mappings for 2D and 3D manipulation.

Hancock et al. (2006) describe a two-finger direct-touch technique for integrated rotation and translation (3DOF control) of a 2D target. Motion of the first finger translates an object, the change in angle of the vector from the first to the second touch determines rotation. They note that scaling can be easily included by evaluating the change in length of the vector.

Moscovich and Hughes (2006) put forward the similarity cursor, a technique for indirect multi-touch control. This two-finger technique allows simultaneous rotation, scaling and translation (RST) for 4DOF control of a 2D target. Cursor position is controlled by the centroid of the fingers, rotation angle is given by the change in angle of the segment connecting the two fingers and the scale factor is delineated from the change in length of this segment.

Nacenta et al. (2009) acknowledge that even with integrated 2D RST controls on multi-touch devices, it is sometimes desirable to only perform subsets of these three rigid transformations. Integrated techniques can be imprecise for partial transformations, since for instance every translation will include a slight rotation and scaling. To alleviate this they discuss three approaches: handles, magnitude filtering and gesture recognition. Their handles technique is based on desktop pointer-based 2D transformation widgets and provides dedicated areas for translation, rotation and scaling only, as well as an area for integrated rotation and scaling. Fully integrated RST transformation with the handles technique can be achieved by using two or more fingers. Magnitude filtering only applies transformations above a certain threshold. Gestural control fits the applied input to the best fit operation by machine learning: translation and scale, translation and rotation, translation only and full RST. Although interaction via the magnitude and gesture techniques can include jumping behaviour as the threshold is passed or the best-fit operation match switches, these techniques achieved the best performance in the user study.

Hancock et al. (2007) present one- two- and three-finger techniques for 5DOF control (2DOF translation plus 3D orientation) in shallow-depth 3D environments. While one- and two-finger techniques require a pseudo-physical model of friction/inertia for 3D transformations (Kruger et al., 2005, see below), the three-finger model allows full independent control of all 5 degrees of freedom: the first contact establishes x/y translation, the rotation of the second around the first enables object rotation around z axis and the 2D position of the third contact is mapped to the remaining axis of rotation. A user study showed that the techniques with the higher number of touches were better both in terms of performance and user preference. This principle is extended by Hancock et al. (2009) to full 6DOF control by using the distance between the first two contact points for z translation. The authors term the direct-touch pin correlation techniques for 3D translation and 1DOF roll *Sticky Tools* and the addition of another finger for further 2D pitch and yaw *Opposable Thumbs* (the analogy to the thumb need not be taken literal—any finger can “oppose” the other fingers, and they can be fingers of another hand).

Inspired by through-the-lens camera control techniques (Gleicher and Witkin, 1992), Reisman et al. (2009) enable integrated rotation and translation using an arbitrary amount of contact points on an interactive surface by maintaining pick correlation. They do this by minimising a quadratic energy function that measures the total squared error between the screen control points and the projections of the corresponding scene points. They calculate an object-to-world space transformation consisting of three translation and three rotation DOF. Under-constrained input allows partial transformation: one finger translates in camera x/y plane, two fingers add camera z axis translation and rotation around camera z axis. Three finger onwards allow integrated 6DOF rotation and translation, they suggest to use the second hand for a more ergonomic use.

Martinet et al. (2009, 2010a) present two variants for direct-touch 3DOF translation control. The *multi-touch viewport* exploits the typical fourway-split screen used in 3D design tools for viewing a scene from three orthogonal views and one perspective view. By 2DOF translation of an object in one viewport with the first touch, the user creates a constraint for manipulation on the other views, in which the user can control the 3rd translational DOF. A line gives visual feedback on the constraint. The *z-technique* can be used with a single viewport and allows translation in screen z: by moving the finger forward (away from the user, usually up on the screen) the object moves away from the camera, backward (toward the user) moves the object toward the camera.

Martinet et al. (2012, 2010b) build on this work in the design of their *Depth-Separated Screen-Space (DS3)* technique for 6DOF control in multi-touch 3D manipulation tasks. This is a combination of the *Screen-Space* technique of

Reisman et al. and the *Sticky Tools* of Hancock et al. DS3 allows both translation and rotation control but separates these based on number of fingers. One finger translates in the plane parallel to the view, a second indirect finger (that does not touch the target) enables z-translation. As soon as two or more fingers touch the target, the same constraint used in the Screen-Space technique controls orientation only.

Manipulation in 3D immersive environments usually employs 6DOF input devices, exactly matching control and object DOF for rotation and translation (Bowman et al., 2004). Problems with reach can be addressed by non-isomorphic mappings that scale input in certain value ranges, or by downsizing the world (so-called world-in-miniature techniques).

2.1.2 Separated Control

When input devices offer less DOF than the object parameters to be manipulated, integrated control is not possible. This is a common problem in desktop interaction for navigating and editing 3D media, since most desktop input and display devices only offer simultaneous control of two DOF. Interface designers thus often face the problem of mapping two control DOF to three to six target DOF.

A solution is to separate control, i.e. split object DOF into manageable subsets (Bowman et al., 2004). With single-pointer input devices, this necessitates a sequential control of such subsets. For 3D manipulation in desktop environments, there are two established approaches to sequential control. The first displays multiple orthographic projections of the scene in one *split screen*. Every view affords translation in the 2 dimensions defined by the view plane and rotation around the view axis, the transformation mode (translate, rotate, scale) is modal and needs to be administered by buttons or hotkeys. The second, *3D widgets*, are spatial handles that are overlaid on top of the target object. By manipulating handles, transformations in one or two dimensions can be performed (for an overview of 3D widgets see Bowman et al. 2004 or Schmidt et al. 2008).

Separated control requires mode control for switching between translation, rotation and scaling mappings. The overhead introduced by mode management can be addressed by multimodal interfaces (Bowman et al., 2004) or by using additional device DOF (e.g. mouse buttons) for fast switching. For instance, Matejka et al. (2009) present whole-hand 2DOF controls for multi-touch input devices where the hand's fingers can provide mode control.

The separation of different types of transformations can also be desirable in order to partially transform an object in a precise manner. By using widgets, filtering input of small magnitude, or detecting gestures, this can be artificially re-introduced to integrated control (Nacenta et al., 2009).

Another solution is to use multiple desktop input devices. Zeleznik et al. (1997) developed several techniques for two-cursor (4DOF) control for partial or constrained operations in 3D, with each cursor controlled by one hand. They enable integrated rotation and translation in a plane: the non-dominant hand (NDH) cursor controls translation, while the second cursor rotates around the axis orthogonal to the plane running through the contact point of the NDH cursor so that pick correlation is maintained. A further technique for 3D orientation is a version of the virtual sphere technique in which the NDH rotates around an axis determined by the object centre and the dominant hand cursor. A third rigid body transformation technique allows translation and scaling in a plane by maintaining pick correlation.

2.1.3 Constrained Control

If high-DOF devices are not available and temporal multiplexing is not desired, interface designers can choose to constrain the interaction to reduce required control DOF. A challenge for designers is that the model behind the constraint must be understood by the user, for instance by basing them on mechanisms already known from other contexts.

Yamane and Nakamura (2003) present a pin-and-drag interface for posing articulated figures. By pinning down parts of the figure, such as the end-effectors (feet or hands) and dragging others, the whole character can be controlled with relative ease. Joint motion ranges, the current joint configuration and the user-set joint constraints (pins) thus allow constrained control of several character DOF with as few as two position input DOF for a 2D character. The various constraints are prioritised so that dragging constraints are always fulfilled and solved by differential kinematics that give a linear relationship between the constraints and the joint velocities.

2DOF controls for 3D orientation make use of the model of operating an affixed sphere such as a globe or a trackball. Virtual Sphere techniques allow control of 3D rotations by using the metaphor of a sphere around the target that can be grabbed and dragged along its surface (Bowman et al., 2004). Dragging within a defined circle radius centrically rotates the object around an axis perpendicular to dragging direction by an amount given by dragging distance. Dragging without the circle rotates the object around the view axis. The *Arcball* technique of Shoemake (1992) rotates objects by drawing shortest-path arcs on the surface of a projected 3D sphere. Start and end point of the 2D cursor are projected onto the sphere and determine the rotation axis as perpendicular to the plane defined by the two surface points and the sphere centre, while the angle is given by the length of the arc. Since only a half-sphere can thus be directly manipulated, angles are doubled to allow up to 360 degree rotation. Although this does not maintain pick correlation, users quickly adapt to the non-isomorphic rotation.

Kruger et al. (2005) establish Rotate 'N' Translate (RNT), a technique for single-point integrated 3DOF rotation and translation control of 2D rigid objects. It is based on a pseudo-physical model of friction that force a rotation during translation of off-centric contact points on rigid bodies, much like moving a real object on a surface with one finger. The more off-centre the contact point the larger the rotation. When the line through contact point and object centre becomes parallel to the direction of movement, the integrated transformation degrades to a translation only. In a comparison of six rotation and translation mechanisms for tabletop interactions by Hancock et al. (2006), RNT is the only one-to-many mapping, potentially controlling 3DOF with only one contact point. Hancock et al. (2007) extend RNT to single-point (2DOF) integrated rotation and translation of 3D projected objects without depth translation (5DOF) The axis of rotation is determined from the point of contact.

2.1.4 Physics-based Control

Several research projects have attempted to leave the world of explicit mappings and enable low-to-high-dimensional control, bimanual interaction and multi-user interaction implicitly by simulating real-world physics. The approaches vary in the user-simulation interaction. These differ from pseudo-physical models such as that used in the RNT technique (see above) in that they use a general physical model rather than one custom-designed for the mapping.

Fröhlich et al. (2000) let users kinematically control intermediate objects that are attached to target objects by springs. They experimented with a number of springs and points of attachment and achieved good results with a four-spring model per hand. This force-based interaction model enables multi-hand and multi-user control in their *Responsive Workbench* system.

The spring attachment is also used by Agarawala and Balakrishnan (2006) to enable interaction with a physically simulated virtual desktop, the *Bumptop*. They only use a single dampened spring for object manipulation in a shallow-3D environment with a pen input device. By enabling users to move objects on the Bumptop much like real objects on a real desktop, the authors hope to afford users to organise their virtual objects in more expressive ways.

Wilson et al. (2008) explore physics-based manipulation techniques. They suggest to represent input by proxy objects that can collide with target objects and exert friction forces. Depending on tracking technology, surface input can have more fidelity than just a set of contact points, e.g. the shape of the whole finger or hand contact area. The proxy objects approximate the dimensions of the whole input feature. Next to persistent simple rigid bodies as proxies they suggest temporary particle clouds to represent contact shapes with higher fidelity. A user study pitched the *proxy* and *proxy particle* techniques against the more established spring-based user-simulation interaction. The spring approach presents a style of interaction closest to kinematic controls and achieved good results in terms of task completion times. However, qualitative evaluation suggested that these kinematic techniques felt more limited, and the likeness to known drag-and-drop behaviours encouraged single-point control. The more dynamic techniques however allowed emergent multi-contact and bimanual interactions styles. Wilson (2009) improves the proxy particle technique to persistent particles, enabling grabbing interactions for physics-based surface interaction. Hancock et al. (2009) extend the idea of physics-based direct manipulation to *Virtual Tools*, mediator objects that extend the range of operations that can be performed on target objects.

2.2 Keyframe Animation

Traditionally, animation has been the business of creating the perception of motion by showing a series of still images in quick succession. This meant that every image had to be meticulously created. In computer-based keyframe animation, only extreme poses or key frames need to be manually established by the animator. Each key frame is edited using manipulation tools, which can be specialised for the target domain, e.g. character poses. Some manipulation tools allow influencing dynamics directly in the scene view. The most common means of specifying editing dynamics is by using global descriptions, such as time plots or motion paths.

2.2.1 Frame Editing

Hand-drawn animation involves drawing a character in a certain pose every frame. The character is essentially created anew for every pose, although techniques such as onion skinning that let the previous frame shine through aid the process. Modern approaches can reconstruct a 3D character from 2D sketches (Davis et al., 2003; Mao et al., 2005), but still require the artist to create a new character image for every frame. In stop-motion animation, poses are defined every frame by manipulating an animatronic figure rather than creating it from scratch every frame. Computer-based animation methods are similar in that a character is created once and subsequently only manipulated. Characters are first created (with box modelling, digital sculpting, or sketching), rigged with a skeleton structure by defining joint articulation and handles, and then interactively manipulated.

Spatial Control

Animation involves editing articulated characters with far more degrees of freedom than single rigid bodies. This is typically performed with separated control through handles and gizmos as in general object editing (figure 2.1). An alternative are skeleton input devices, mechanical armatures with location and orientation sensors, with a structure matching the virtual character. Esposito et al. (1995) built the Monkey, a mechanical controller in the shape of a human. It features 35 DOF plus eight binary switches for control actions. Knep et al. (1995) built an articulated input device in the shape of a dinosaur. This armature was used for stop-motion style frame-by-frame animation of virtual dinosaur creatures. The authors claim that physical input and virtual rendering

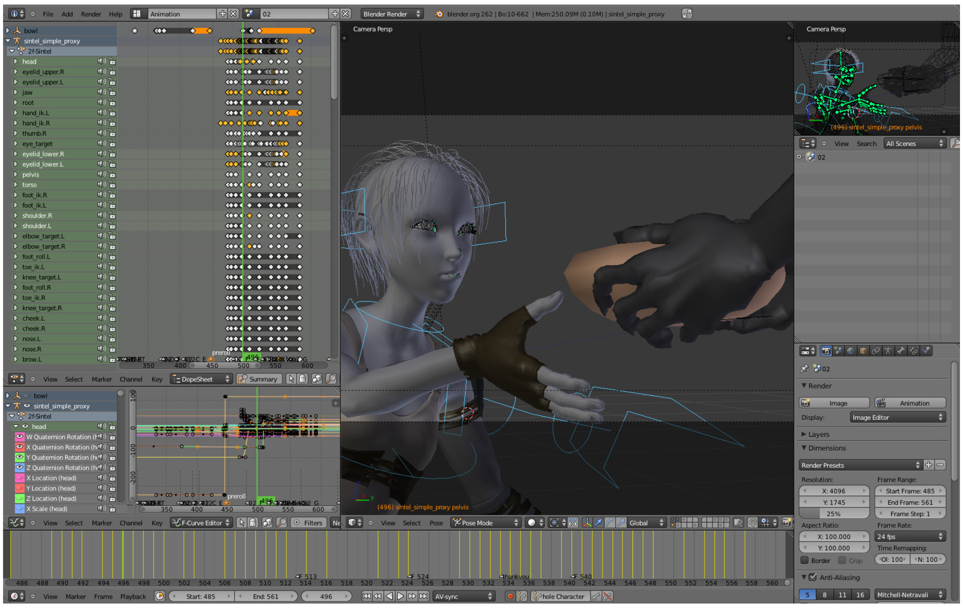


Figure 2.1: Example of an interface for frame-based animation, featuring a 3D view with manipulation handles (centre), and various time plots: dope sheet (top left), graph editor (centre left), timeline (bottom). Animation material attributed to the Durian Open Movie Project, licensed under CC BY 3.0.

make for the best of both worlds, and the one-to-one mapping between device and character allows for an easy use. Where character and armature joints do not fully match due to software, animation and physical constraints, they define a hierarchy of joints by setting end-effectors and leg joints as anchors, an approach similar to that used in motion data retargeting (section 2.3). A simple curve matching procedure that is initiated at anchor joints then matches the digital joint layout to the physical. For the feature film *Jurassic Park*, individual armatures for different types of creature had to be built to maintain the direct mapping. While this theoretically enables the simultaneous control of all DOF (provided enough puppeteers operate the armature), technical and practical constraints often result in a sequential procedure. For these reasons, skeleton input devices are used for frame-by-frame animation or to define keyframe poses, rather than for interactive puppetry (section 2.3.2).

Spatial editing between keyframes can be achieved indirectly by editing interpolation functions or by defining a new key pose. Alternative approaches abstract from the underlying (key)frame representation. Motion warping (Witkin and Popović, 1995) allows the global editing of motions independent of the underlying original specification. Interactively editing the pose at a certain frame changes a warping curve rather than the original motion parameters. This warping curve describes a deformation of the original motion parameter that can be applied it. The benefit is that this smooth deformation preserves the fine structure of the original motion. Gleicher (1997) create motion displacement maps from a more general formulation of space-time constraints. Optimisations in the constraint solver allow interactive editing of constraints in a direct manipulation style, such as positioning a joint at a specific frame, controlling foot plant and handhold locations, or location of obstacles to avoid. Lee and Shin (1999) further improve interactive frame-based motion editing via displacement mapping. Instead of solving a complex optimisation problem (Gleicher, 1998), they break it down into subproblems. With an efficient B-spline approximation that minimises local approximation error they are able to edit motion data with motion warping by displacement maps. Problems of motion warping resulting from too few knots in the displacement map are addressed by a hierarchical approach. An analytical solution for inverse kinematics further increases the algorithm's performance to allow interactive (offline) editing by direct manipulation. The techniques proposed by Snibbe (1995) also abstract from underlying motion representation by describing motion as a function of the arc length of the motion path over time. These allow direct manipulation motion editing at arbitrary instants in time via displacement maps over a specified time window. Snibbe remarks however that these tools only work if the underlying motion curve has sufficient control points within the edited region, since otherwise the curve can behave non-intuitively.

Temporal Control

While time is usually controlled through global descriptions (section 2.2.2), some techniques enable timing by directly operating on the visuals.

Snibbe (1995) suggests timing techniques that do not require time plots but can be administered by directly manipulating the target or its motion path in the scene view (figure 2.2). Within a specified interval, the animator drags the target along its motion path at a certain point in time. Temporal displacement functions temporally translate the object's motion graph in the specified interval so that it reaches the manually specified location at the current time. Snibbe further describes a direct manipulation velocity control technique, in which dragging operations from the target along the motion path do not change the object's position but alter the velocity at this time instant, creating speed-up or slowdown effects at the current position and time. The position of the object at the current time and the duration of a specified editing interval are not changed. As with spatial editing, the practicality of temporal editing with displacement functions depends heavily on the underlying keyframe distribution.

Timing by direct manipulation in the scene view is also supported by the latest animation software packages. Tweaking motion trail handles allows for temporal instead of spatial translation; visual feedback can be given by changing frame numbers adjacent to the handle. *Timing beads*, special point-based handles along the path, can be shifted in order to control the velocity at that point. As suggested by Snibbe, tick visualisations along the curve give visual feedback on the temporal transformation.

Spatial control of time has also been proposed for video navigation. Kimber et al. (2007) propose reach-through-the-screen video time control by grabbing and moving objects along their trails. Video analysis determines object tracks, which are also overlaid in the interface. They propose a distance metric for interactive setting of the time based on the mouse cursor position. It involves the weighted components of spatial distance, temporal distance, and a penalty for changing time direction in order to enable smoother navigation through time.

Karrer et al. (2008) put forward a similar direct manipulation time browsing interface that allows dragging of video features through time (figure 2.2). They use image processing to detect feature trajectories in digital video. To ensure smooth propagation through time they extend the distance metric that matches cursor position to feature position by a temporal component (frames).

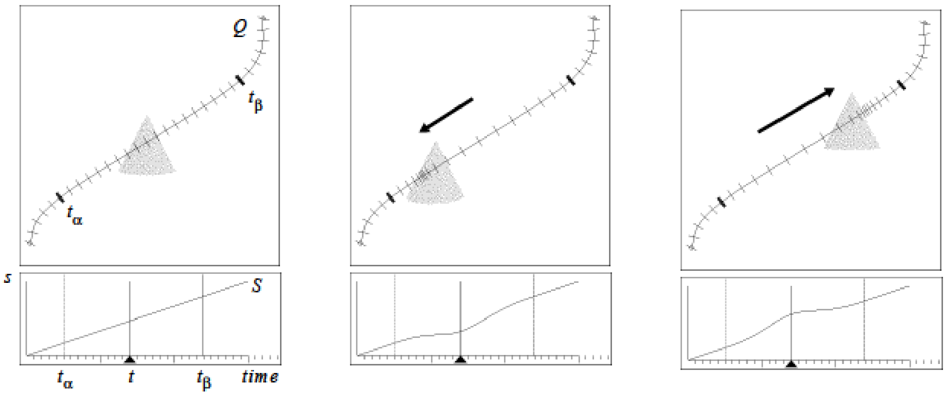
Goldman et al. (2008) present techniques for annotating and navigating video, as well as video to still image composition, using video motion analysis. They discuss several applications of the object-based approach, including direct manipulation video browsing, which uses a simple spatial distance metric. They suggest to incorporate time into the metric in order to deal with discontinuous jumps when solving the space-time mapping is ambiguous. Building on this they also demonstrate a constraint-based video control that uses several inputs and solves for the closest match in the video feature trajectories.

Dragicevic et al. (2008) propose feature-based direct manipulation time browsing (figure 2.2) using a proximity metric for pointer-based dragging that includes trajectory arc length, making it time-independent. Next to ensuring arc-length continuity in browsing, they add a penalty term for directional changes in order to maintain directional continuity.

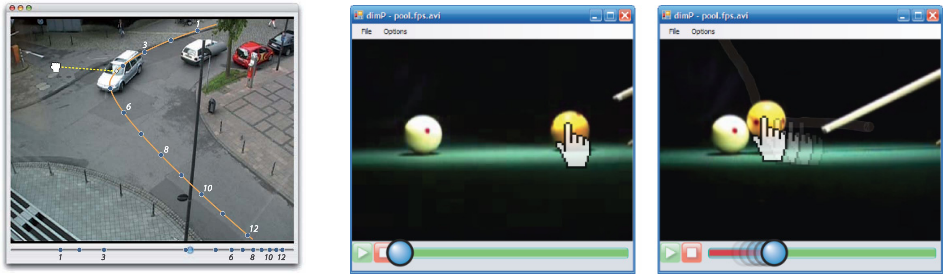
Shah and Narayanan (2011) use the direct manipulation for video editing, for instance to retime a segmented part of the video against a still background or the rest of the scene.

2.2.2 Describing Dynamics

Every animator works with some specification of dynamics. Aids for describing inter-frame relations were already used before computerisation, in the form of camera exposure or dope sheets, paper forms for specifying which animation frames should be shown on which camera frame. Computerisation has brought powerful techniques for displaying and editing motion dynamics globally. Common graphical dynamics descriptions are *time plots* that spatially represent the timing of animation events or the dynamics of spatial parameters (figure 2.1), and *motion paths*, which can be valuable aids in visualising and editing motion.



(a) Snibbe (1995)



(b) Karrer et al. (2008)

(c) Dragicevic et al. (2008)

Figure 2.2: Object-based time control interfaces. Images taken from source.

Time Plots

The exposure sheet or dope sheet is an essential device in hand-drawn animation, which was developed in the early days of animation (Thomas and Johnston, 1981; Taylor, 1996). With the dope sheet, the animator specifies the timing of individual frames one at a time. There is a line for each frame, a typical dope sheet will have 96 lines or four seconds of film (Williams, 2009). In the horizontal columns the animator specifies the occurrence of frames for a particular element in a scene, one column per element. There are further columns for dialogue, sound and music cues, camera, and notes. This essential device has transferred to modern computer-based animation tools. In the digital form, the dope sheet has the horizontal and vertical dimensions swapped, to align it to the panoramic layout of computer screens as opposed to the portrait orientation of paper. In keyframe animation, the dope sheet remains an important device for specifying the timing of elements of the scene, as animators can transform individual or groups of keyframes in order to only change the timing of motion.

Object or character parameters can also be mapped against time in a 2D function graph. Before computer-aided animation was practised at economical scale, Baecker (1969) already envisioned static spatial representation of dynamics in the form of plots of animation parameters against time. Two such waveforms, one plotting the x-coordinate of an object versus time, the other the y-coordinate, could for instance completely describe two-dimensional motion. These could be edited or redrawn (Baecker's GENESYS system supported tablet-based pen input for motion and curve sketching) to redefine the temporal or spatial component of the motion, or both. Editing dynamics with static data plots is a common tool for computer animation artists today. Motion graphs are two-dimensional plots that map feature values (vertical axis) against time (horizontal axis). With a 2DOF input device, such a graph thus allows integrated, simultaneous spatiotemporal control. In keyframe animation the motion editor is the standard way to manage keyframe value interpolation, typically by means of Bezier curve handles.

Some approaches abstract from the keyframe representation. Time warping or time remapping is a common feature of current commercial animation software. By manually defining a spline via keyframe-like handles/constraints, the animator specifies a time function that relates the current timing to a new desired timing (Witkin and Popović, 1995). A common method is to manipulate a graph spline that maps source to target time by adding and shifting control points on the curve until the desired result is reached. This allows the efficient timing regardless of actual motion representation (keyframes or capture data). Similarity search techniques that identify similar motion sequences to the manually edited one can further increase efficiency by propagating temporal edits to the similar sequences (Mukai and Kuriyama, 2009). Guided time warping can also be used to propagate the timing properties of the modified segment to the remainder of the clip by using the edited segment as the example (Hsu et al., 2007). Possibly the best way to visualise time remappings are two-dimensional plots of source against target time. McCann et al. (2006) use such plots for representation of physics-based automatic retiming. Users can then make adjustments via this graph interface to control the solver by adding, removing or adjusting constraints and interactively viewing the result.

Time plots are usually employed to edit motion that was already created—for instance a sequence of roughly plotted character poses in straight-ahead animation. However, time plots can also be used to create animation from scratch. The changing of a value over time can be specified by directly creating a function graph of value against time. Sketching such function graphs is already suggested by Baecker (1969) for the GENESYS system, and modern animation packages also support motion curve creation directly in the graph editor. In the animation sketching system of Mao et al. (2005), the trajectory of a character can be determined by sketching motion paths against time per spatial dimension.

Path Descriptions

A more literal visual representation of global dynamics are motion paths. Current animation packages allow visualisation and editing of a whole motion sequence via motion paths, sometimes called motion trails. The motion path typically depicts the exact trajectory of a selected feature. Motion can be edited by manipulating the motion trail like a three-dimensional spline. For editing the general trajectories of characters, Gleicher (2001) suggests to use motion paths at different levels of detail. For instance, although a character might actually waddle sideways during an otherwise forward motion, a straight path might be a better representation for most editing purposes. Instead of using splines, motion path editing can be posed as a more generalised optimisation problem for satisfying spatial constraints. Kim et al. (2009) and Lockwood and Singh (2011) use an as-rigid-as-possible deformation algorithm (Igarashi et al., 2005a) that finds the best solution for these constraints in a least squares sense.

Motion can also be interactively created with global movement descriptions. These can define the specific change in translation of a rigid body (also its orientation if the input device supports such degrees of freedom), or a general description of motion of a more complex articulated figure (Gleicher, 2001). If used in the latter manner as a guide for the motion, inverse kinematics or gait generators can be employed to define animation details (Balaguer and Gobbetti, 1995). Static path descriptions cannot convey timing information, this must be added in later steps or recorded along with the interactive specification of the path description (see section 2.3).

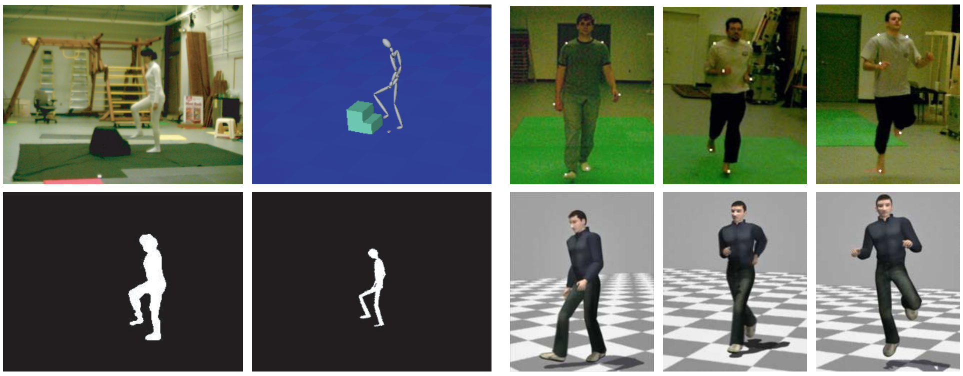
The system for biomechanically-inspired motion path editing by Lockwood and Singh (2011) performs automatic time warping after each interactive path edit to preserve relationships between path shape and velocity based on biomechanics. Their reasoning behind this is twofold: First, often users will just want to manually adjust the spatial aspects of motion and will just want the timing to “look right”. Second, the spatial and temporal attributes of motion are tightly coupled and should not need to be modified independently.

2.3 Performance Animation

Motion capture digitises the live performance of an actor or puppeteer by tracking a number of key points in space over time and combining them to obtain a representation of the performance. The recorded data then drives the motion of a digital character. The entire procedure of applying motion capture data to drive an animation is reached is referred to as *performance animation* (Menache, 2011). In a typical setup, an actor’s motion is first recorded, then the data is cleaned, processed and applied to a digital character. Since the digital character can have quite different proportions than the performer, retargeting the motion data is a non-trivial task (Gleicher, 1998, 1999). In this form of performance animation, capture and application of motion data to an animation are two separate processes, data handling is done *offline*. *Online* performance animation immediately applies captured data to a digital character, creating animation instantly. Computer-generated imagery driven by the real time motion capture data can be rendered and displayed immediately to audiences and performers. This allows to create animation on-the-fly, allowing the performer to react to the audience or other performers. It is used in settings where offline performance animation cannot deliver, such as live broadcasts or interactive exhibits (Sturman, 1998; Gleicher, 1999). Furthermore, the live rendering can be fed back to the performer, allowing him to "continually regulate their performance to achieve the desired visual result" (Sturman, 1998). This visual-motor feedback loop allows for more flexibility in the actor-character mappings. The continuous feedback in real time allows non-literal mappings, meaning that the target's motion need not mirror that of the performer (Oore et al., 2002a). Processing limitations entail that performers can often only see a low-fidelity pre-visualisation of the final rendering since details of the performance-character mapping cannot be evaluated in real time (Menache, 2011). Even if the final result will require adjustment and production, instant feedback to the performer can be very useful (Gleicher, 1999). Thus, online performance animation can also be valuable for offline production.

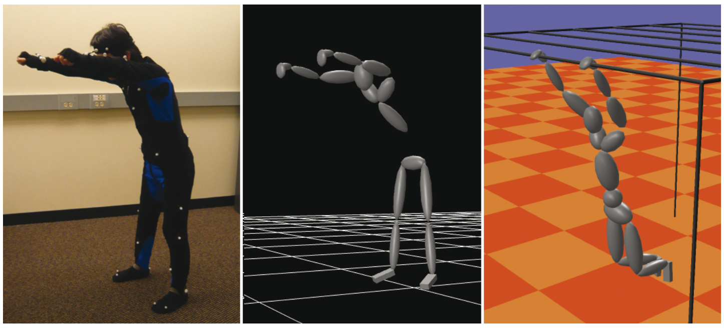
A further factor in performance-based motion creation is the style or metaphor of control. This is characterised by the degree of abstraction or indirection between the human actor and the virtual counterpart. Bodenheimer et al. (1997) distinguish between mappings that attempt to reproduce human motion as literally as possible and highly abstracted mappings, which are primarily concerned with the character of the motion and only secondarily with its fidelity and accuracy. Many performance animation efforts aim to represent human motion accurately and limit the abstraction to a minimum. Motion capture performers, usually people trained at bodily expression such as actors or dancers, use only the senses with which they have learned to act, (e.g. kinaesthetic and proprioceptive feedback) and do not rely on further feedback on the mapping. They thus embody the character by slipping into its virtual skin like into any other role, performing *virtual acting*. This is in contrast to performance controls that work with more complex relations between performer and character. In this approach the created motion only indirectly relates to the author's motion, and the style of interaction is more akin to puppetry: animators or puppeteers learn "to manipulate the figure indirectly as a puppet, rather than as a direct representation of themselves" (Bodenheimer et al., 1997). Such a style of interaction is very much dependent on instant visual rendering of the result. Just as traditional puppeteers would rely on mirrors or camera feeds to adjust their performance, *computer puppetry* requires instant renderings of the applied input to allow performers to adjust their motions.

The rest of this section presents the related work in the categories of virtual acting and computer puppetry, before discussing performance control systems that cannot be easily related to either.



(a) Lee et al. (2002) match 2D camera image silhouettes against rendered silhouettes of motion data.

(b) Chai and Hodgins (2005) match tracking data from sparse reflective markers to a motion database.



(c) Ishigaki et al. (2009) combine kinematic tracking data with dynamically simulated data in real time for realistic interaction with virtual props.

Figure 2.3: Virtual acting systems. Images taken from source.

2.3.1 Virtual Acting

Performance animation with full-body control involves setups with a high correlation between the performer and the digital character. Real time mappings either use high bandwidth devices for coordinated control of all character DOF, or employ models based on example data or a physical simulation.

Integrated Control

Typically, an actor's "joints are mapped directly ("literally") onto the corresponding joints of the character" (Oore et al., 2002a), which requires tracking setups that can process a large number of DOF.

The first device for full-body motion capture used in the entertainment industry is the rotoscope. Invented in the early 20th century, it projected live action film footage onto the animator's worktable frame by frame, allowing him to trace shots of real people for realistic motion (Menache, 2011). The modern day equivalent are optical, electromechanical or electromagnetic motion capture systems that track the motion of markers or sensors placed along the performer's body. In all cases, the raw capture data needs to be processed in order to obtain translational and rotational data of an articulated skeleton. While this can be done online during capture by electromagnetic or electromechanical systems, complex marker setups still require offline processing for optical tracking (Menache, 2011).

After cleaning the raw capture data, the motion is applied to a digital character. This is not a trivial task, especially when accurate representation of human motion is the goal, which is another reason this is usually done offline (Bodenheimer et al., 1997). If actor proportions differ grossly from those of the character, the data must be adapted in order to fulfil constraints such as foot ground contact or interaction with other features, a procedure called *retargeting* (Gleicher, 1998). A significant problem of motion capture data is that it consists solely of motion samples captured at high frame rates, making editing extremely difficult. A great deal of work has been done to transform capture data into more manageable formats, such as functional representations (Balaguer and Gobbetti, 1995; Sudarsky and House, 2000).

Choi and Ko (1999) present a procedure for retargeting motion data in real time. Instead of minimising an objective function subject to space-time constraints (Gleicher, 1998), they propose to calculate the joint configurations of the target with closed-form solutions to inverse kinematics based on the Jacobian matrix. While the space-time approach requires integration over the whole interval and is thus intrinsically offline, their inverse rate control algorithm operates only locally, optionally allowing on-line control. Thus, an actor can control a character with different proportions online, benefiting from immediate feedback on the performance.

Example-based Control

Full-body control can also be enabled with more sparse tracking information. This requires mappings that are learnt from correlating sparse input data with existing animation data. These usually limit the range and style of motion that can be created online, and such techniques are on a thin line between synthesising new motion from existing motion, and editing existing motion.

Lee et al. (2002) developed a method that maps low-dimensional control data to unlabelled, unstructured motion capture data for the interactive control of virtual characters. For this they propose a two-layer approach. The lower layer approach models transition probabilities between any two motion capture frames in a markov model. The higher layer clusters motion frames, a tree structure determines which cluster can be reached from each frame. They test their approach with three different interfaces. The choice interface allows the user to interactively select the next cluster from a low number of options.

A sketching interface selects frames so that the character follows a trajectory by introducing an additional cost factor for closeness to a sketched path. A vision-based interface matches user silhouettes against precomputed rendered silhouettes from the motion database (figure 2.3). They see potential for the selection and sketch interfaces for applications limited in desired speed and environment complexity. They attribute the most complete control over the character with the vision interface, however their approach results in a 3 second delay making it far from interactive.

Yin and Pai (2003) propose using a pressure sensor mat as an input device. They first record desired body motions and align body capture data with the according pressure sensor readings. During use, online readings of the sensor are matched with the prior recorded motions using a distance function based on 10 features of the sensor image. A motion editing step ensures that played back motions match temporal and spatial parameters of the input by performing inverse kinematics and time warping on the recorded motion data if necessary. With fairly simple algorithms they achieve recognition rates of around 80% and a near-real time performance with 1 second delay.

Chai and Hodgins (2005) propose a method for controlling a high-DOF character with low-dimensional control data in the form of a sparse set of retro-reflective markers (figure 2.3). In a first offline stage of their procedure they fill a database with motion data captured with a high-DOF marker-based system. During runtime, performer motion is tracked with only a small subset of seven markers. The database is searched for examples that are close to the current sparse signal. These samples are used to calculate a local model from which the current character pose is determined. They claim a quality of results comparable to those of a commercial full-body tracking system, and point out the advantages of less time to gear up the performer, less intrusion and less cost.

Vögele et al. (2012) devise a means to steer mesh animations online with sparse motion capture data. During a training sequence, input motion is correlated with existing mesh animations to create a linear model for pose and shape. Both motion capture data and mesh sequences are analysed for pose information, which is then parameterised. The optimal mappings between input and target shape and pose parameters are calculated as regression functions for each limb pair of the input and output skeleton. The authors demonstrate this approach with offline motion transfer between captured quadrupeds and bipeds as well as interactive steering of mesh animations.

Physics-based Control

Limitations in the motion capture system or the performer's physiology to produce certain desired motions can be overcome by simulating parts of the body and their interaction with the environment.

Ishigaki et al. (2009) combine real time full-body motion capture data, physical simulation and a set of motion examples to create character movement that a user can not easily perform, such as swinging, climbing and swimming (figure 2.3). The virtual environment contains predefined interaction points such as the handles of a monkey bar or a rope. Once the character's end-effectors are brought into proximity of an interaction point, control changes so that the character motion is no longer fully controlled by the motion capture. A simplified simulation that treats the intentional contact as a universal joint connected to the character's centre of mass by a linear dampened spring enables the calculation of the overall dynamics of the character. Further, character pose is synthesised as an intermediate between online motion capture data and offline motion examples, depending on wether end-effectors are constrained or unconstrained on user and character. In order to detect the correct user intention, the similarity of the user's current pose and example motion is computed in a lower-dimensional space derived from principal component analysis of pose data. The synthesised pose is integrated with the conditions of the simulation to combine final character motion. Their system not so much maps low-level tracking data to higher-level character data but rather enables creating motion that is difficult or impossible to act out directly.

Nguyen et al. (2010) integrate online full-body motion capture and a physical simulation in order to allow complex virtual interactions. The character's behaviour is determined by blending a kinematic controller and a dynamic controller, the former operating on joint positions, the latter on torques. While both components strive to the pose determined by online capture, the former is not influenced by external forces. By manually fine-tuning the blending weights for the two components, the authors generate believable results in complex scenarios such as interaction with a tethered ball, a balloon and in a tug of war.

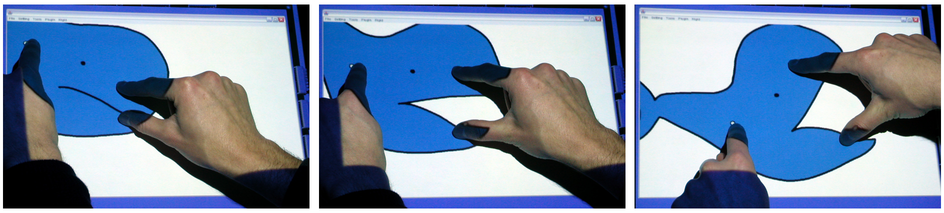
2.3.2 Computer Puppetry

For performance animation of stylised or non-humanoid characters it is desirable to control them in a less literal fashion. Such a style of performance control is often referred to as computer or digital puppetry (Sturman, 1998). Since the relation of performer movement and character movement is less literal, it is largely dependent on live feedback on the result. The central problem is that even simple characters have many degrees of freedom to control. Solutions can be divided into continuous controls that use spatial multiplexing, temporal multiplexing or constraints, indirect controls, controls using existing motion data, and controls based on a simulation of real world physics.

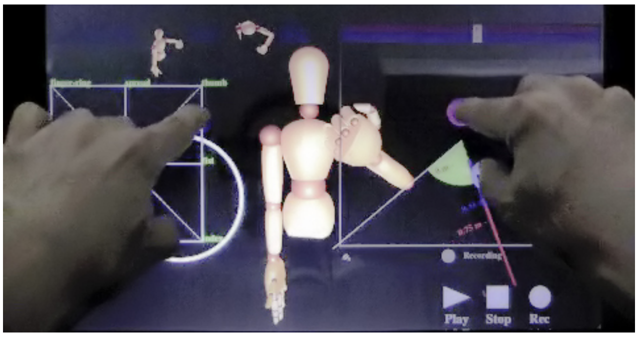
Integrated Control

For optimal control, it is desirable for the input space and the character space to be maximally congruent. The dimensionality of the output space dictates this somewhat: 2D animation is less demanding since it lacks the 3rd dimension.

(a) The system of Dontcheva et al. (2003) explores layered acting and implicit editing by mimicking existing motion with a 6DOF widget.



(b) Moscovich et al. (2005) use their 2D rigid body deformation algorithm for multi-touch puppetry.



(c) Kipp and Nguyen (2010) propose bimanual touch-based controls for the control of an articulated arm.

Figure 2.4: Computer puppetry systems. Images taken from source.

One of the first approaches to digital performance animation is the GENESYS system of Baecker (1969). This employed a stylus input for "clocked hand-drawn dynamics, or the dynamic mimicking of animated behaviour" in order to produce lifelike, energetic movements. Informal tests, in which individuals with varying degrees of artistic skill and animation training constructed short cartoon sequences with the help of GENESYS, asserted the descriptive tools a "rich, expressive, intuitively meaningful vocabulary for dynamics".

Based on interviews with expert animators and beginners, Davis and Landay (2004) define basic animation tasks, and infer controls that are partially nonlinear and part real time. In their accessible animation system K-Sketch (Davis et al., 2008), motion is created by dragging targets with a stylus in recording/playback mode. An evaluation showed good results regarding ease of use and natural interaction, the concern for lack of controls for precise manipulation was also raised.

Moscovich et al. (2005) explored multitouch interactive surfaces for computer puppetry. They developed a mesh deformation algorithm (Igarashi et al., 2005a) based on spring models that allows physically plausible movement and deformation of arbitrary 2D graphics. With multi-point interaction devices co-located with the display, this allows direct-touch moving, stretching and squeezing of 2D virtual puppets (figure 2.4).

Barnes et al. (2008) present a video-based computer puppetry system for users of all skill levels. It uses self-drawn paper cutouts that can be animated using a camera setup. The system tracks paper puppets in real time and composites them onto a different background with the puppeteer's fingers removed from the image. Articulated puppets can be created in their puppet builder submodule, and for ease of control individual segments can be separated during capture, while they are rendered as one articulate figure. Additional effects that make use of the digital medium are scene change trigger puppets, 2.5D setups for perspective effects based on the horizontal position of the puppet, automatic walk cycles and additional environment rendering effects such as puppet-directed shadows and rain and snowfall overlays. Their approach leverages the common experience everyone has with paper cutouts and the proprioception afforded by physical objects. Since input is planar and content 2D, a direct mapping of puppet location and orientation is possible.

High-bandwidth input devices are the only option for live performances including only a single puppeteer. In his overview of computer puppetry production in television and film, Sturman (1998) gives the example of Mike the Talking Head presented at the Siggraph 1989 electronic theatre. Mike's facial expressions, voice and head movement were all performed by a single puppeteer. However, it is more common to have more than one puppeteer. The devices used for production settings are a combination of full body tracking and pedals, gloves, joysticks, and further custom-built devices such as a mechanical arm for jaw movement (Sturman, 1998). In current productions of animated series, computer puppetry is used as an efficient and comparatively less expensive alternative to keyframe animation (Jurgensen, 2008).

Even when input and output degrees of freedom match, physical interdependencies of input DOF can still limit a mapping. In full body tracking, the joint locations are dependent on actor size and body proportions. If the performer's proportions significantly differ from character proportions, this can lead to problems with the character interacting with objects in the scene, such as the floor or props. This problem is well recognised for motion capture, where the process of retargeting can often lead to such difficulties, and several off-line solutions to this exist (for an overview see Shin et al., 2001). Shin et al. (2001) recognise this problem for interactive setups and propose an approach that maps input based on importance. Their importance measure uses a few simple heuristics and mainly considers the distance of end-effector to an object in the scene. For instance, proximity to an object gives the location of an end-effector a higher importance that maintaining the orientation correspondence of connecting joints. Their system has been demonstrated in two Korean TV shows that feature live animation of a virtual character that can interact with presenters shot live in the studio.

The greater the performance range of a character, the higher the demands on the puppeteer. One way of meeting these demands is to employ several puppeteers that split the performance and thus the degrees of freedom of the character between them. Common setups for simple characters in live performances are two puppeteers, one for the body and the other for facial expression, and three for recorded performances. The capabilities of motion capture devices further influence these decisions. Humanoids are fairly straightforward to map, while controls for multi-legged or no-legged creatures must be given some thought. In practice, budget limitations influence these decisions as much as other factors (Sturman, 1998). The multi-touch interface for animating deformable drawings of Moscovich et al. (2005) places no constraints on how a user grasps and manipulates a drawing. Their system easily incorporates multiple users, and the authors explicitly suggest that two users may collaborate to coordinate control of different parts of the drawing, reflecting the potential of horizontal interactive surfaces for collaborative work (Kruger et al., 2003; Scott et al., 2004). Nguyen et al. (2010) even suggest remote collaboration with their virtual reality performance animation system. In their setup, however, each performer controls an individual character, and they do not discuss coordinated control of non-humanoid characters or facial motion. The video puppetry system of Barnes et al. (2008) also supports multi-user control, either co-located, with two or more users controlling complex single or multiple puppets, or remotely, with each individual puppeteer seeing the results of all characters merged into a single scene. With the system of Ninomiya et al. (2008), individual puppeteers can control a virtual marionette by computer vision-based finger tracking at their desktop. Multiple puppeteers thus control a puppet and the audience can join in remotely to form a networked virtual marionette theatre.

Separated Control

For live performances, control needs to be addressed with high-bandwidth input devices or performers acting in parallel. With recorded performances the puppeteer has more options. Capture sequences or just parts of them can be retaken, or slightly modified, and complex motion can be built up in passes. Layered or multi-track animation allows the performer to concentrate on only a small amount of action at a time and create articulated motions step by step.

In the 1996 film *The Adventures of Pinnochio*, the computer-generated character of the cricket had two pairs of arms. For performance animation of this character, the two pairs of arms were captured in two passes, first the upper set, and then the lower set (Sturman, 1998).

Oore et al. (2002a) employ layered motion recording for controlling subsets of a characters's DOF. For the animation of a humanoid, they divide the character DOF into three parts and animate these sequentially: Two 6DOF devices are used to control the motion of both legs, both arms, and torso and head in three passes.

Dontcheva et al. (2003) make motion layering to the principle of their live animation system. By controlling a 3DOF widget (figure 2.4), the animator can control different character features in subsequent passes, layering the animation track by track. They distinguish two variants: layering over multiple features, i.e. recording torso, arm, leg movement, and layering over a single feature, where an initial coarse motion can be refined over subsequent layers. For the latter, they suggest three feature-interaction paradigms: absolute, trajectory-relative and additive mappings. Their control system thus allows motion creation and, to a certain degree, spatial editing of created motion. The authors further propose a technique that eliminates the need for explicit feature selection. With the implicit editing technique the animator mimics an existing motion in a first step and then performs the desired adjusted motion in a second, which is then applied to the animation.

The system of Neff et al. (2007), which controls character pose with low-dimensional mouse input via correlation maps, makes heavy use of layered animation. In their system, every loop of input is recorded on a separate layer, which can be turned on or off at will. It also offers overlays, which are layers that are driven by the input of a previous layer but use a different correlation map. This allows automatic synchronisation with previously recorded movements and saves the performer from having to synchronise new input with prior motion.

Svensson et al. (2008) describe an interface for live animation recording and mapping based on sequencer interfaces known from digital audio editing. The sequencer interface is made up of data tracks that let the user specify which input to map to which output. Apart from a general description the authors do not elaborate on mapping design or editing and layering functionality of the sequencer approach. User feedback showed imminent feed-forward information to be crucial for the anticipation of events, and the authors suggest data stream visualisation and foreshadowing as two examples.

The video puppetry system of Barnes et al. (2008) also supports layering for controlling individual puppets or different features of an articulated character. Re-recording individual features overwrites previous motion recorded for that feature.

Constrained Control

If high-bandwidth input or sequential control are no options, it is possible to map lower-dimensional input space to higher-dimensional scene space by constraining target DOF. Such many-to-one mappings can take advantage of constraints in the target domain. For instance, human biomechanics features frequently occurring synergetic joint motions or known symmetry and phase relationships (Laszlo et al., 2000). To increase flexibility, constraints can be altered during control (Neff et al., 2007), or user-generated (Kipp and Nguyen, 2010).

Neff et al. (2007) present a performance-based system that uses a 2 DOF input device (the mouse) to control a 33 DOF humanoid character. For this they propose to use correlation maps to define high-level mappings. During usage, an animator can switch between these maps using hotkeys or GUI menus.

Kipp and Nguyen (2010) propose a multitouch interface for simultaneously controlling the many degrees of freedom of a human arm (figure 2.4). The authors solve this with a bimanual interface. The dominant hand controls 3D hand position and by extension the arm through inverse kinematics. Next to surface position for x and y positioning, the pinch of two fingers can be used for controlling the z depth. The arm swivel is defined by the absolute angle of the line that the two touch contacts create. This maps 4DOF input control (finger contact centroid x and y, distance of index and thumb contact, rotation of the line through index and thumb contact) to a 6DOF arm by exploiting biomechanical constraints in human arm movement, especially the elbow joint. The same finger orientation technique is used to control palm rotation with the non-dominant hand. The position of the non-dominant index finger on a morph map determines hand shape.

Yamane and Nakamura (2003) suggest that their pin-and-drag interface for interactive specification of motion constraints can also be used to "edit motion in motion". This allows to edit parts of the motion, such as the trajectory of an end-effector, in a puppetry-like manner, while the motion is otherwise maintained as closely as possible.

Indirect Control

It is often desirable to abstract between input and output even further. Indirect controls trade expressive power for ease of use and increased learnability. They make up for lack of definition through input by drawing control from motion libraries or programmed generators. With indirect controls the tool user hands over expressive power to the tool designer, who has considerable influence over the end result in designing pre-defined motions or programming motion engines.

Oshita (2005) present a pen-based interface for interactive control of a human figure. Online pen input is recorded as a series of points. Based on the spatial and temporal difference of consecutive points, they categorise pen gestures into one of four types: moving, stop, jump and reset. Depending on further parameters such as duration or direction of pen gestures, sub-categories such as turning, right/left steps and directional jumps are determined. A set of parameters is delineated from the input which can include the pen inclination and pressure (depending on the category), and is passed to a motion generation engine that blends motion capture examples. Weights for all the example motions of the category are calculated by combining linear and non-linear coefficients of a radial basis function. A small user study shows their interface to be superior to a baseline gamepad interface in terms of accuracy but not always in terms of speed and reaction time.

Input can be simplified to interactively defining key positions of significant events, such as foot contact. In this manner, discrete control can for instance describe a walking animation. Kolhoff et al. (2005) use a graphics tablet capable of concurrently sensing two pen inputs for a performance-based walk animation interface. Foot orientation is given by a pen's angle of inclination, which the tablet sensors are able to detect. From foot contact positions and orientations they derive a walking animation with a simple gait generator: Keyframes for a foot are added at the start and end of a footprint, tangent data is added to lift the feet off the ground between footprints. A keyframe for the pelvis is created for each footprint, its timing in relation to step can be determined globally. While pelvis height is fixed, the other spatial parameters are calculated as weighted averages of the corresponding values of the feet. Lockwood and Singh (2012) conducted a study on the potential of hand gestures for performance-based character gait animation and devised a prototype interface for walk animation through finger walking on an interactive surface. Asking test subjects to imitate reference walking, running or jumping motions with their fingers, they found that finger gestures have a large discrepancy to presented motions regarding factors such as velocity and ground contact. They conclude that subject's performances were more illustrative in nature and propose a gesture-based interface. Their prototype retrieves six features from a user's performance and matches these against baseline finger performances for a set of target motions using standard machine learning techniques. The appropriate animation is then selected from a set of predefined motion clips and edited so that it matches the spatial parameters of finger motion: the enacted motion path, animation scale and number of cycles. A second user study showed classification rates of up to 74% and high user satisfaction ratings.

Johnson et al. (1999) coin the term sympathetic interface for puppet-like control of a virtual character via a doll "device". The control interface consists of a plush toy with embedded sensors for sensing pressure, motion and orientation of parts of the doll. It was modelled after the design of the virtual character in an interactive cartoon story installation. In an early prototype, the authors experimented with direct mappings but found these to make the character seem robotic and lifeless at times. Further, they wanted it to stay "in character" regarding motion style and behaviour. Thus they developed *intentional* control that recognised simple gestures using machine learning with hidden Markov models. While this allowed them to map a variety of inputs to pre-animated character behaviours, the intentional control did have weaknesses in character navigation, where most users seemed to be used to more direct control from video games.

Example-based Control

Mappings can also be defined using pose or motion samples. This procedure requires a set of pose or motion examples that cover the desired range of performance. These essentially constrain target space, supporting lower-DOF input devices. As opposed to merely blending samples from motion libraries, such mappings still allow the creation of new movement based on an example-defined control space.

A way of using low-dimensional signals for interactive control of articulated characters is to define a set of complex (extreme) poses and to map these to spatial targets. A variety of motion can then be achieved by moving a cursor between these targets. The proximity to each target determines the interpolation weights with which the current pose is calculated from the set of predefined poses. Kipp and Nguyen (2010) use a morph map of nine distinct arrangements of the fingers of a hand to control hand gestures of an articulated arm. The hand shape is defined by the position of the index finger of the user's non-dominant hand on a matrix of nine hand poses displayed on an interactive surface while the dominant hand manipulates the arm. The spatial keyframing technique of Igarashi et al. (2005b) works similarly: the animator defines key poses offline and associates each with a 3D cursor position that corresponds to that pose. These targets are visualised as small spheres. During runtime the character pose is synthesised based on the proximity of the input cursor to the spatial keyframes. While this allows easy interactive editing, the scope of motion is limited to the morph space defined by the key poses as well as the spatial arrangement of the targets. Laszlo et al. (2000) use a similar technique where poses are desired character joint configurations for proportional-derivative controllers in a physical simulation. Their GUI displays six predefined poses in two rows and three columns. Moving across this matrix, the mouse cursor's x coordinate controls blending between three states of forward propelling leg stances (columns) while the y coordinate controls leg extension by blending between extended and crouching versions of the three leg poses (rows). Assuming a fixed phase relationship among the four legs, a single mouse can simultaneously effect coordinated control even of a quadruped. Just like the hot-switchable correlation maps of Neff et al. (2007), they propose discrete keystrokes to switch between multiple continuous controls, for instance to switch between swing control of a character's both legs.

Lam et al. (2004) control walking motion with a finger walking interface. The movement of index and middle finger is tracked by a data glove that records finger flexion. In a learning phase, the animator mimics with his fingers a recording of a walking motion. The index finger joints are related to the left leg and the middle finger joints to the right leg by mapping functions. These are constructed by matching tips and pits in the leg joint and finger joint trajectory plots. Sample points between these matched trajectories are then fitted to B-splines that define the mapping function. In this way, a recorded walk gait can be changed to a running or hopping gait, synthesising new motion from recorded data. The authors remark that the mapping function only works properly when the newly generated motion is similar to the original motion. Walking samples could thus not be edited to synthesise a two leg jump, for instance. Essentially, their system allows for performance-based editing of motion data.

Physics-based Control

Regardless of how literally they relate performance to target, controls for computer puppetry discussed up to here are of kinematic nature in that they transfer real world motion or events onto a digital target mechanically. Physics-based controls strive to make interaction richer with partially or globally simulated dynamics.

Laszlo et al. (2000) lament the lack of realism in performance interfaces and claim that this limits the animator's ability to identify with the character. Their system lets the user interactively control a physical simulation. They demonstrate how such user-in-the-loop techniques can create bipedal motion using only a mouse: one spatial dimension exerts torque on the leg joint, the other on the knee joint. Mouse control is supplement with discrete keyboard control actions that trigger a sequence of forces exerted for compound joint movements.

Oore et al. (2002b) develop local physical models to supplement 6DOF input in their human puppet control interface. Each thigh bone is controlled with one 6DOF device, while knees and ankles are driven by physical forces reacting to motion applied to bones further up in the hierarchy.

Zhao and van de Panne (2005) propose two interfaces for controlling an articulated character in a physical simulation. The first interface divides every character motion in the simulation into several stages which are each associated with a target pose that can be triggered by discrete control events. For this they present the action palette GUI with virtual buttons that can be clicked with the mouse to trigger the associated action. The relative x, y position of the mouse cursor on the button determines two parameters that characterise the action, such as angle or duration. In the online version of this interface, the timing and position of discrete mouse click events determines the spatiotemporal aspects of the simulation. In an offline editing mode, action timing can be adjusted on a timeline slider and the two parameters per action can be edited. The second interface maps body motions such as crouch balance and height and waist bend and twist to continuous input from gamepad joysticks to drive a snowboarding simulation. Anecdotal evidence from informal tests indicates that the gamepad interface is more challenging than typical games interfaces but also allows more freedom of expression.

Shiratori and Hodgins (2008) use the three-dimensional accelerometers in Nintendo WiiMote devices to control a physically simulated character. The interface is designed so that the user performs motions similar to those desired with two WiiMotes. High-level qualities of the input motion determine the character's motion controllers, which were implemented as proportional-derivative controllers, or change their parameters. The phase of the two signals determine the gait of the motion: in phase motion activates the jump controller, out of phase motion the walk or run controllers, depending on signal frequency. WiiMote inclination changes the characters orientation, no motion activates the default stepping controller. A study pitting three variants of attaching sensors to the user against a joystick interface showed that the performance interface outperformed the baseline regarding completion of test tracks and user ratings.

2.3.3 Other Performance Control Systems

Some performance control systems can not be easily associated with the concepts of acting or puppetry. They make use of metaphors such as paths, timelines or triggers, use adaptively evolved, or entirely arbitrary mappings.

Evolved Control

Instead of designing performance controls by hand, Gildfind et al. (2000) propose to let the puppeteer create mappings in a process of experimentation and refinement. For this they develop the adaptive performance control system APECS which uses genetic programming to evolve mappings via user-assessed fitness. For any combination of input device and puppet model they start with a randomly generated set of samples across the search space. These are then evolved by letting users assign fitness values to members of the population—each member of a generation of evolved mappings. The evolution can be short-cut by starting with a well-defined set of preferred gestures, only control systems which transform these gestures will then be considered. An important requirement for the genetic programming approach is that users maintain consistent feedback through the assigned fitness values. In custom-built control systems, designers will design mappings based on experience. In the end however, the user will often have to adapt to the interface to a certain degree. The advantage of the evolutionary approach is that "the constant feedback between user and system helps ensure that the resulting control system is well customised to the user's preference and skill" (Gildfind et al., 2000): each puppeteer can start out with the control sequences he favours and map them to a desired output motion. The authors further point out that the search across the whole configuration space might produce mappings that designers would never have envisioned.

Path Control

Interactive path descriptions are the most abstract form of performance animation. The animator describes a 2D or 3D trajectory as a general description of the desired motion. Depending on the mapping, such dynamic path descriptions can give the continuous position (and with 6DOF input devices orientation) of a specific feature, or a global trajectory of a more complex target. The path is drawn interactively, giving the animator a primary feedback on his input. In the case of 2D paths a preferred input device is the stylus, as this evokes the animator's skills of drawing on paper. Path-based control requires completing a single or a sequence of paths to determine a motion. The earliest time at which an author can receive feedback is after he has completed input—control is never real time interactive. This puts such approaches in contrast to puppetry interfaces that provide immediate visual feedback. These can also make use of pen-based input for sketch-like interaction (Oshita, 2005; Kouda et al., 2009), but rather use this form of input for continuous spatial control.

Path descriptions are an important technique in the sketch-based animation system GENESYS, developed by Baecker (1969). These continuous movement descriptions are proposed along with static movement descriptions (selection between individual animation cels) and rhythm descriptions as three forms of global dynamic descriptions. In GENESYS, path descriptions can be sketched as static plots of the change of a single spatial coordinate against time, or dynamically sketched in real time by spatially mimicking the 2D motion and recording the sketch and its timing.

Terra and Metoyer (2004; 2007) suggest to split the spatial and temporal design into two separate phases of the animation process. While spatial arrangement of key poses can be defined using conventional means in the first phase (section 2.2), the temporal layout of the keyframes can then be performed online in the second. For this they propose a technique with which the user sketches the desired motion path of the animation to be timed. This path is then matched to the actual trajectory by determining and matching salient spatial features of both curves. The timing in which the curve was sketched, sampled along with the spatial input, is then used to determine the new time for each keyframe. The performer can fully focus on the animation, mimicking or acting it out similar to computer puppetry. On the downside, the imperfect input matching process is not interactive and often requires user input to resolve mismatch. Thus the animator does not get feedback on the result until after the capture take. Mao et al. (2005) use this approach to time motion trajectories in their sketch-based animation system.

The virtual reality animation system of Gobbetti and Balaguer (1995) also allows performance animation via high-level path descriptions. With a 6DOF tracking device a path is specified along with orientation. The animated objects interpret the path as a goal depending on object primitive: while rigid bodies exactly replicate the path, the motion of articulated bodies is computed by inverse kinematics and virtual humans are animated via a gait generator. Balaguer and Gobbetti thus understand motion paths as non-literal representations of motions, as proposed by Gleicher (2001) for static motion editing.

Thorne et al. (2004) propose a system that combines path descriptions with a gestural language for determining gait and motion style. By sketching symbolic gestures into the motion path, the animator can trigger character actions such as jumping, leaping or different walk styles. The system determines the type of action by matching gesture input to a set of defined gestures. The exact motion is synthesised by blending samples for each gait and motion and adapting them to sketched path and timing.

The motion sketching interface of Popović et al. (2003) allows users to sketch the trajectories of simulated rigid bodies. In this approach, the animator specifies a rigid body's trajectory with the mouse or an instrumented object. This "motion sketch" can have physically impossible motion and arbitrary timing—the animator specifies the sequence of a motion, not its dynamics. An optimisation then estimates parameters of a physical simulation that best replicates the specified motion. The result is a clean physical simulation with realistic motion and timing. In the puppetry variant, their interface allows the specification of motion by demonstration by manipulating an instrumented object. The example of Popović et al. is a tracked real pen that is puppeteered to produce an animation of a virtual pen jumping and landing in a cup.

Abstract Control

Kouda et al. (2009) propose a performance animation system that maps the two axes of a line drawn with a pen on a tablet to any two transform parameters in real time. The position of the pen in each spatial control dimension determines the output values for the two transform parameters, such as translation along z axis or rotation around y axis. To allow simultaneous control of a third parameter, the arc length of the curve drawn for the first two parameters is mapped to the x-axis of a new coordinate system and the y axis to the third parameter, allowing constrained 3DOF control. Instant visual feedback on the result supports learning of abstract mappings. A small user study showed that this approach could be grasped by novice users.

Timeline Control

Sampath (1999) employs timeline scrubbing, a common technique for video browsing, to retime an animation. As users interactively control playback time in real time, they implicitly create a time warp mapping from playback time to real time. When releasing the scrub action, the time warp is applied instantly. This provides fast results for a task that otherwise takes much longer in offline editing, e.g. by shifting keyframes manually, and is potentially easier to use. The scrubbing approach uses a well-known device, the timeline, and offers users interactive feedback on their actions since the animation is updated as the input occurs. On the downside, there is no relation between the horizontal scrubbing action and the spatial configuration of animation motion.

Trigger Control

Some works have touched upon online timing with discrete, non-spatial input. Zhao and van de Panne (2005) offer a retiming mode for their physics-based animation system with discrete input control. After initial motion creation has laid out the ordering of simulation events, a retiming mode lets the user act out the motion with a sequence of timed spacebar presses. "Each spacebar press then specifies the revised timing of the next event, with the events always preserving their original ordering" (Zhao and van de Panne, 2005). The authors claim this to be a useful way to quickly explore multiple variations of the same motion with slightly different timings. Terra and Metoyer (2007) discuss clicking input devices such as the mouse for performance timing. The salient points on the motion path (or otherwise determined significant points) could thus be triggered with non-spatial trigger commands, using only the timing of the input events to create a time warp. The authors argue that not only does this seem less intuitive than their proposed sketching technique, it further loses velocity information between keyframes which could be used to adjust animation curves for effects such as ease in or ease out.

2.4 Non-interactive Animation

Further animation methods complement keyframe and performance animation. Programmed motion is a powerful tool for efficiently creating large-scale animations with high realism. Powerful simulation packages such as the products from NaturalMotion[1] create virtual actors and stuntmen based on models and rules rather than human performance or interactive manipulation. With increasing amounts of high quality motion capture data available, a lot of work is dedicated to analysing, processing, modifying and reusing data. These are not the tools of choice for an artist wanting to express his vision of a character's behaviour, or move objects in unrealistic ways. They are also less reliant on a working interface between the artist and the subject matter. For these reasons this review touches only select works in non-interactive animation in order to illustrate this vast area of research and application.

2.4.1 Simulation

Perlin (1995) describes a notation and method for scripting character behaviour. For each type of action, per joint they define extreme poses and periodic interpolation between them, as well as noise functions for added realism. These actions can be weighted and thus blended for simulated character behaviour. Rules for communication between multiple agents can create believable character simulations.

---

[1] `http://naturalmotion.com`, last accessed 6th January 2019

In the research of Hodgins (1998) on simulating human motion, the human body is simplified to an articulated skeleton of rigid bodies. Control systems for simulating human movement are implemented as state machines. Within each state, control laws determine the exact movement of each part of each limb. This can be implemented by proportional-derivative controllers that determine the current joint torque required to reach a desired angle. States and control laws are based on biomechanics literature.

Treuille et al. (2007) discuss a procedure for interactively synthesising character behaviour from a range of motion clips by specifying high-level goal parameters such as character gait and orientation. They use these as well as other parameters (including obstacle position and speed) as a low-dimensional representation of character behaviour, on which they define cost functions. With a near-optimal policy defined on four controllers their system is able to react to user changes in these high-level desired parameters interactively.

Liu et al. (2005) provide a new approach for determining the parameters of a physical simulation of human movement. They propose nonlinear inverse optimisation to estimate physics parameters from motion capture data. Their approach treats the motion capture data as an optimal solution to an optimisation problem with unknown parameters and known constraints. This allows them to maintain the style and characteristics of the motion capture performer and generate physically realistic new motion in that style.

2.4.2 Working with Motion Data

Wang et al. (2006) present the cartoon animation filter, which processes motion data that does not have typical cartoon animation characteristics to fulfil the animation principles of anticipation/follow-through and squash and stretch. This is achieved by feeding a smoothed, inverted and time shifted version of the acceleration back into the motion signal. While this does not produce the same quality as motion effects hand-crafted by animation experts, it does provide a simple tool that can be easily applied to a large range of applications, for instance motion previews.

Mukai and Kuriyama (2009) propose a timeline interface augmented with visual motion abstracts (such as character poses) for transferring kinematic and temporal properties from an example to a target motion. They incorporate only example-based spatial editing because, as they claim, manual operations often lose the naturalness of complex operations. Their technique transfers joint rotation mean and variance (the latter corresponding roughly to motion intensity) or end-effector trajectories from example to target within a specified time range. Their system further allows edit propagation. This applies the edits not only to the selected frame range but also to ranges with similar motion.

For the "motion analogies" approach of Wu et al. (2008), differences between two given sample motions are calculated. This difference can then be applied to a third motion to transfer motion style. In their example, the difference between a walking and a jogging motion is calculated and applied to a crouching motion to produce a jog-like crouching. By treating motion sequences as space-time curves, they are able to apply methods from high-dimensional curve processing.

Hsu et al. (2007) provide an alternative to manually specified time warps—guiding time warps by a reference motion. By manually specifying sparse key time constraints and supplying a reference motion the target motion is retimed to resemble timing characteristics of the example. The time warp is formulated as a discrete time compression of an input, other desired warps can be achieved by suitable transformations of the input. Local velocity and acceleration estimates are computed for each frame and the objective function is formulated as distance to equivalent velocities and accelerations in the reference clip. By formulating the optimisation as a dynamic programming problem, the authors are able to elegantly compute the optimal time warp from all candidate time warps.

Mukai and Kuriyama (2009) propose a similar temporal alignment based on discrete time warping. Their pose timeline interface further supports drag-and-drop operations for coordinating joint or end-effector timing. Thus one set of motion curves is temporally shifted in relation to another so that their phase relations match those of selected features in the reference motion.

2.5 Video Games

Video games have a strong connection to animation. Most modern video games make heavy use of animation in order to breathe life into the game world. In this sense, games are one application area amongst many others, such as film, television, or education (section 2.5.1). But animation is also created *with* and *in* video games. The actions taken by players and the responses of the game world constitute a form of motion design, often conveying a story. This is most evident in game genres where players control characters in a virtual world, like a puppeteer controls puppets (section 2.5.2). In *machinima*, the art of 3D game-based filmmaking, animation and video games ultimately come together to form a novel means of creating animated movies (section 2.5.3).

2.5.1 Animation for Video Games

Animating for video games differs significantly to animating for film or television. Linear media require that the animator only create the specific behaviour of characters and objects required by the story, and this only needs to be viewed from a single camera angle. In interactive media such as video games, behaviour and view are spontaneously defined by the player. The animator cannot foresee the decisions of the player, which is why he must create animations for all possible player actions that must meet certain criteria of completeness and realism so that they work in combination with other animations and can be viewed from every angle. Such motion libraries contain elementary animation sequences can then be looped, blended and combined in real time by the game engine (Kelland et al., 2005). They can be created with keyframe, performance, or non-interactive animation. The prevalent method used in modern 3D video game productions is performance animation, due to increased quality and lower costs (Watt and Policarpo, 2003). By interactively directing pre-defined animations, players thus essentially perform a kind of digital puppetry with indirect control.

2.5.2 Between Gameplay and Puppetry

The most evident example of gameplay as animation are games that involve directing an avatar through a game world. In these, character control is central to gameplay, which is also the central task of character animation. In part due to constraints of input hardware, in part due to the desire for intentional rather than direct controls, interfaces for such games control high-level parameters such as direction and speed of movement, while lower-level character motion is determined by predefined animations or motion routines. High-DOF input devices can extend the degree of motion control, further blurring the lines between gaming and puppetry. As players are able to influence more character DOF, their possibilities for expression are increased. However, while all games use some form of motion capture, few offer the extent of motion editing required in animation practice—if a player is not satisfied with his performance, he will have to do it again. Most games lack techniques for even the basic task of time control. Notable exceptions are the titles *Prince of Persia: Sands of Time* (Ubisoft, 2003), *Zeit*^{2} (Brightside Games, 2011) and *Braid* (Number None, Inc., 2008), in which the player must navigate time as well as space. Yet while these games incorporate time control in innovative ways, they do not provide the degree of editing required for animation.

The connection between interactive character control and gaming is also evident in scientific literature on performance animation techniques. Section 2.3.2 discussed diverse mappings for interactive character control, which often double as game controls. The pen-based interface for control of a human figure of Oshita (2005) is typical for a game interface in that control is executed on a high level (via sketched paths) and then transferred to the motion of a biped by means of a gait generator. The baseline interface used in their comparative study is a standard gamepad. In designing their plush toy-based character control interface for an interactive installation, Johnson et al. (1999) discuss trade-offs between intentional (triggering pre-authored character motions with gestures) and direct control, pointing out that while their variant allows more influence of the designer on the style of character behaviour, it does create an indirection that can irritate users. Zhao and van de Panne (2005) explore user interfaces for controlling dynamically simulated characters in the context of animation and games. Works discussed in section 2.3.1 make use of full-body motion capture. In many approaches live input is mapped to existing motion capture data, limiting control (Lee et al., 2002; Yin and Pai, 2003; Chai and Hodgins, 2005). The works of Ishigaki et al. (2009) and Nguyen et al. (2010) integrate kinematic input and dynamics simulation for realistic character control, providing a glimpse into the future of motion control in video games.

2.5.3 Machinima

Machinima is the art of filmmaking in 3D virtual environments. The term is a combination of the words machine and cinema. Using ‘i’ instead of ‘e’ was an unintentional misspelling, which was kept as a nod to animation and Japanese anime (Marino, 2004; Kelland et al., 2005). Machinima started in the 1990’s as gamers recorded in-game action in order to demonstrate their gaming prowess, and distributed these shorts in the community. Since games provide complete worlds with easy controls, enthusiasts were able to quickly create recordings of animated characters acting in detailed environments with this approach. Game levels became virtual sets and gamers became puppeteers that controlled virtual actors (alternatively, virtual actors can also be scripted or directed by the game’s artificial intelligence). Early machinima used in-game assets, and explored topics and themes specific to the gaming community. With the beginning of the new millennium, productions increasingly left their gaming roots behind them, and more and more short (and few feature-length) movies were made with a focus on artistic expression and storytelling. Machinima was soon hailed as the new way of making animated movies at comparatively cheap cost (Marino, 2004; Kelland et al., 2005). The advent of online virtual worlds, massive multiplayer online role-playing games, and the communities emerging around these further boosted the movement, as more and more people used this kind of medium. Film-makers profited from the in-game market for assets to bolster their virtual props and sets (Dellario, 2011).

Tools for creating machinima are closely tied to a specific game engine. Early tools did not amount to more than simple recording/replay functionality in the game, while more mature tools provide non-linear editing, and more complex scripting and/or puppeteering functionality (Marino, 2004; Kelland et al., 2005). These are often the same tools that game developers use to create in-game cut scenes between interactive play, also called cinematics.

Using game engines for animation or virtual filming has benefits as well as limitations. Modern 3D games provide a complete game world with physics, animated models, and special effects while offering comparatively simple controls for puppeteering game characters. This gives authors a lot to build upon, as opposed to other methods where animations must be created from scratch. The limitations lie in the dependency on the game developers and their short product cycles. Also there are copyright issues involved in using third-party game engines and assets. In order to eschew such problems, machinimators have variously created their own models, animations and even engines from scratch. However, the more effort is put into "traditional" modelling, animation and development for custom machinimation, the less the initial advantage of rapid, low-cost production applies (Marino, 2004; Kelland et al., 2005).

Computer puppetry in games remains limited, as is any performance control interface that merely activates and blends pre-defined animations. Ultimately, the requirements for gaming and expressive animation simply do not overlap enough to further the marriage of gaming environments and animation tools.

2.6 Discussion

To conclude, the contributions to ongoing research made by this review are pointed out, its limitations are discussed, and the next steps on the research agenda are considered.

2.6.1 Contributions

The comprehensive overview of related work on motion design interfaces spans literature from computer graphics, HCI, artificial intelligence, tangible computing, mixed reality, virtual reality, and entertainment computing. Despite coming from varying backgrounds, the work discussed was conducted in order to make computer animation more efficient, expressive, or accessible. To the knowledge of the author this is the first collection of this body of work strewn across many computer science disciplines.

The discussion is "normalised" by maintaining a perspective on the interaction aspects of related work. Individual papers and products are shaped by views, opinions and terminology of their respective discipline, which can influence everything from problem definition to approach, analysis of results, and entire presentation. The consistent focus of issues of motion control makes the benefits and drawbacks for the user more evident and individual contributions more comparable. While this is straightforward for some cases it requires a closer look at the mechanisms at work in others.

It is a first approach to structuring the vast body of work on motion design interfaces with an integration of a method view known from computer graphics (Hodgins, 1998) and a view concerned with aspects of user control. This addresses divergent views in the main disciplines of computer graphics and human-computer interaction and can aid the understanding of how control styles and animation methods relate. It also provides first clues to characterising motion design interfaces, which is pursued in chapter 3.

2.6.2 Known Limitations

In the presentation of related work the underlying computational considerations are treated secondary to issues of interaction, which stands contrast to the typical perspective taken in graphics literature. For instance, the categorisation of motion editing tools by Gleicher (2001) into those based on inverse kinematics, signal processing, constrained optimisations and motion paths is largely orthogonal to the structure chosen here. These approaches stand side by side without any considerations as to how they could be connected.

Furthermore, the review abstracts from input/output devices to a significant extent. Devices are often interchangeable, and a hardware-centric approach is prone to getting lost in an analysis of device limitations, losing the bigger picture. Of course, this is less helpful for application developers faced with the technical details of a specific device. However, device abstraction helps understanding the principles behind the hardware and software configurations of interfaces, which is the appropriate level for this thesis. Analyses of input devices (Card et al., 1991; Bowman et al., 2004) or performance control hardware (Menache, 2011) can be found in the literature.

2.6.3 Research Agenda

The discussion of the state of the art brought forward a large variety of digital tools for creating the illusion of life. The interaction-centric view alerted us to several characteristics of motion control interfaces such as the tasks involved in motion design, degree of simultaneous spatial control, the underlying model or metaphor of an interface, and the general relation of artist and medium space and time, as well as first indications of how these figure into the user experience. The next chapter investigates this design space of motion design tools further in order to arrive at a framework with which to better analyse qualities and deficiencies of state of the art systems.

Future work could analyse the interplay of interface characteristics and underlying algorithms and mathematical models, which could further benefit cross-disciplinary work. Also, an analysis of the design space of hardware for which these techniques are implemented could offer more concrete insights for interface designers and hardware manufacturers.

Chapter 3

Computer-based Animation: An HCI Perspective

Interactive computer animation involves a dialogue between a human—the animator, actor or puppeteer—and the application, through a machine—the computer with which character behaviour is edited or a performance is captured. The degree of interaction ranges from constant feedback loops in frame-based animation to the machine passively recording actor performance.

Given the importance of interaction for most computer-based animation methods, it is still only treated as a topic subordinate to issues of algorithms for modelling, representation and rendering. While there is an increasing trend in computer graphics research to consider the needs of the artist (e.g. Popović et al., 2003; Schumacher et al., 2012), the dominant focus is on challenges of modelling and representing graphics and the involved computational problems. On the other side, HCI research has not yet approached the complex field of animation interfaces to a significant degree. As a result there exist no conceptual frameworks for analysing interaction issues of current animation tools. A similar case has been made for information visualisation, where interaction plays a large role in analysing visualisations of complex data but has yet received little scientific attention (Yi et al., 2007).

An HCI perspective on computer animation methods and interfaces is needed. This chapter makes a first step by constructing a design space of user interfaces for spatiotemporal media. In the course it is discovered that in prior work there have been no attempts to sufficiently describe the relations between the real space-time of the user and the virtual space-time of the medium. Yet these relations are fundamental to interactions with continuous visual media. It is not only shown that various categories of space-time interaction exist, but that they can be taxonomised based on the relations between the spatial and temporal components of user input and the spatial and temporal dimensions of a continuous visual medium. This framework will be the basis of investigations into new animation interfaces in the following chapters.

The design space of animation interfaces is constructed in section 3.1. A new taxonomy for spatiotemporal interactions is developed in section 3.2. Section 3.3 discusses contributions, limitations and next steps.

3.1 Design Space of Animation Interfaces

For theoretical and practical work on motion design interfaces a framework to structure discussion is desirable. For such a purpose it is common to use a design space, which structures the designer's options in creating an artefact by identifying aspects that influence the creation process. These can be called the dimensions of the design space. There is no established procedure for generating design spaces—the only criteria is that it serves theorists and practitioners as a reasoning device in analysis and design.

Existing interface design frameworks can not be readily used for animation interfaces. High-level frameworks are concerned with too general categories (Frohlich, 1992; Nigay and Coutaz, 1993; Jacob et al., 2008; Döring et al., 2013) and thus have limited descriptive power for animation. The approach of Card et al. (1990, 1991) only analyses input devices but not their mapping to output. Frameworks are often established for a specific application domain—e.g. information visualisation (Card and Mackinlay, 1997; Yi et al., 2007)—or task domain—e.g. navigating digital media (Karrer, 2013)—to a varying degree of overlap with animation issues. It is thus necessary to generate a new design space through a combination of deliberation on basic principles and accordances in existing interface taxonomies.

Whether viewed at the level of interaction techniques (Bowman et al., 2004; Hinckley and Wigdor, 2012) or participating agents (Card, 1989), literature defines human-computer interaction as such: a *human* accomplishes a certain *task* via a *machine*. Task, human and machine can thus serve as the basic design dimensions (figure 3.1). Task decomposition is frequently used to structure interaction techniques (Foley et al., 1996; Bowman et al., 2004; Hinckley and Wigdor, 2012). Regarding the human, one can look at cognitive and physiological aspects. A cognitive approach to interaction involves the central notion of the metaphor underlying an interface (Neale and Carroll, 1997), which is also often used to structure techniques (e.g. Bowman et al., 2004). When considering design tasks such as animation, the physiological (and cognitive) aspects of how humans apply their hands in manual tasks is central (cf. Buxton and Myers, 1986; Kabbash et al., 1994; Leganchuk et al., 1998). Regarding the machine, it makes sense to abstract from the underlying hardware and consider more generally the mapping between human input and machine output. For manipulation interfaces this involves considerations of spatial and temporal distance between input and output, the relation of input and output morphology and the input bandwidth. These are addressed respectively by directness, correspondence and integrality, which are discussed throughout the literature

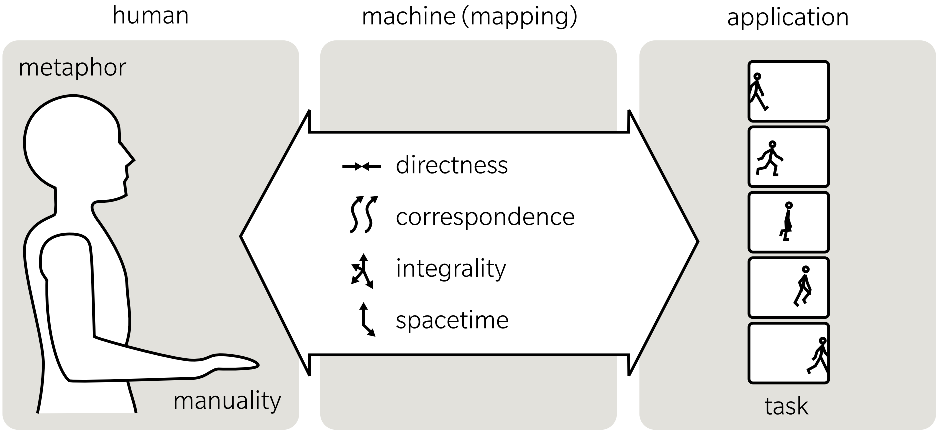


Figure 3.1: The design space of animation interfaces.

(sometimes under different terminology), and have been integrated into one model by Beaudouin-Lafon (2000).

These six dimensions, task, metaphor, manuality, directness, correspondence and integrality, constitute a suitable framework for discussing interfaces for spatial interaction, as they comprehensively cover the factors involved (figure 3.1). They are described in greater detail in sections 3.1.1–3.1.6. What they do not cover is an essential part of motion design, time control. For this reason a seventh dimension treats space-time relations in the mapping (section 3.1.7).

3.1.1 Task

Tasks determine the goal of interactions and the purpose of tools, and are thus the first and most important thing to consider. Interaction techniques can structured based on a *decomposition* of the tasks that can be accomplished through them. One can also consider the diversity of tasks that the same technique can handle, its *versatility*.

Task Decomposition

Task decomposition analyses and groups techniques based on the purpose of the action. By breaking down tasks into constituent parts further and further, one arrives at atomic tasks that become meaningless if further decomposed. Such atomic tasks have been termed *basic* (Foley et al., 1996), *canonical* (Bowman et al., 2004) or *elemental* (Hinckley and Wigdor, 2012) interaction tasks. For example, Foley et al. (1996) collect five basic interaction tasks (BITs) used for most applications: positioning, orienting, selecting, entering text, and entering numeric values. BITs can be combined to more complex tasks that have been termed *composite* (Foley et al., 1996) or *compound* (Hinckley and Wigdor, 2012) interaction tasks. These are combinations of BITs integrated into a unit, such as dialogue boxes, construction tasks, and manipulation tasks. Whether a task is basic or composite is a question of definition—for instance, Bowman et al. (1999) propose a task-based taxonomy in which they decompose selection into subtasks, making it a composite rather than basic task. A solution is to make this device-dependent: for instance, the 2D positioning task is an elemental task with a mouse, but consists of two subtasks when x and y coordinates are quantified separately (see also section 3.1.3).

The three main composite interaction tasks identified for *spatial control* are *target selection*, *manipulation*, and *navigation* (Bowman et al., 2004). For *temporal control* one can equally identify the task of determining *time instant or interval*, *temporal access control* tasks for navigating time-based media (reverse, fast-forward, or browsing), and *temporal transformations* for temporal editing (scaling, cueing, inverting or translating) (Little, 1994).

A high-level distinction of tasks in computer animation is whether motion is created from scratch or existing data is being modified. *Motion creation* techniques vary with the animation method. In computer-aided frame-based animation, the animator constantly switches between spatial manipulation and temporal navigation: a key pose is specified, a new frame is selected, a new key pose specified. Animators will ever so often review their work and edit it spatially (at certain keyframes) or temporally (by shifting keyframes in time) thus intertwining creation and editing tasks. In performance animation, the tasks of creation and editing typically differ significantly: *Changed wording to clarify*motion is created by capturing a performance and mapping it to a character, which usually involves further editing of the capture data (Gleicher, 1999). Procedural techniques generate motion by simulating physical and biomechanical processes, the parameters of which are given by the animator. Independently of how animations were created, *motion editing* is often necessary to adjust motion data to new circumstances. "The common goal of motion editing is to make a desired change to existing motion data while preserving the essential quality and features of the original motion as much as possible" (Kim et al., 2009). These can be spatial and temporal alterations. Different creation methods have an effect on the structure of motion data, impacting editability. For instance, while parameterised animation data (e.g. keyframes and interpolations) lends itself well to editing for a skilled animator, motion capture is made up of a large amount of data points that are not practical to edit manually. Significant amount of research has been done on parameterising motion capture data for better editing (see section 2.2). To change simulations, settings need to be adjusted and the simulation re-run. Some work has been done on enabling more direct user control to specify and edit procedural animations (e.g. Popović et al., 2000). A subtask of motion editing is *timing*, which changes an animation's timing without affecting poses. Such time warps can be either specified manually (Witkin and Popović, 1995), on the basis of examples (Hsu et al.,

2007) or a physical model (McCann et al., 2006), or by performing the desired timing (Terra and Metoyer, 2004).

Versatility

Generality (Schöning et al., 2009) or versatility (Jacob et al., 2008) characterise the variety of interaction tasks that can be performed with an interface. This can range from supporting a large amount of tasks from varied application domains to only supporting a single, domain-specific task.

To illustrate the generality dimension of their taxonomy for graphical user interfaces, Schöning et al. classify typical WIMP desktop systems as specialised for 2D interaction and less suited for 3D interaction, while immersive virtual reality interfaces are specialised for 3D rather than 2D interaction.

Animation can require the control of very diverse creatures and objects. Performance controls are traditionally very specialised, with the constraint often already beginning at the hardware level in the form of full-body motion capture suits or special hand-puppet input devices (Sturman, 1998; Jurgensen, 2008). Yet research has also brought forward more general controls, such as the 2D multi-point deformation technique of Igarashi et al. (2005a). Physical armatures for stop motion animation also have the constraint built in to the device, restricting their use to the type of creature that they were built for, such as humans (Esposito et al., 1995) or dinosaurs (Knep et al., 1995). Pose editing tends to virtual instruments for rigid translate, rotate, and scale transformations. While these are specialised for one type of manipulation subtask they can be flexibly used for any type of target.

3.1.2 Directness

Directness characterises the mental and physical "distance" between user and the target domain. Based on prior definitions (Shneiderman, 1983; Hutchins et al., 1985), Frohlich (1997) defines two dimensions to directness: *engagement* and *cognitive directness* (figure 3.2). Engagement describes how close the user's mode of interacting is to the target. Frohlich associates this with the mode of interaction metaphors conversation (low engagement) and manipulation (medium engagement) This can be extended by embodiment, which is characterised by the user identifying with the target rather than operating on it (high engagement, see also section 3.1.5). Cognitive directness is the mental and physical distance to the target. Frohlich further defines design criteria for cognitive directness in conversation (familiar terminology, natural language and personal relevance) and manipulation (coherent real-world metaphor, natural actions, continuous representation). He complements these with criteria for *interactional gracefulness*: responsive visualisation for manipulation interfaces and short rapid turns, mixed initiative and explicit repair for conversation interfaces.

Zhai and Milgram (1998) develop a similar continuum for manipulation, from using tools (abstract, indirect) to direct engagement with the target (isomorphism). They establish three criteria for directness:

1. simple transformations from input to output (absolute mappings are preferable to relative ones),
2. a control-display ratio of one, and
3. minimal location/orientation offset between control and display space.

Beaudouin-Lafon (2000) supports the distinction between instrumental and direct manipulation and gets quite literal in his definition of directness for manipulation as the *spatial offset* between input and target (the literal distance, cf. Zhai and Milgram's criteria 2 and 3) and *temporal offset* between input and response (cf. Frohlich's responsive visualisation). The latter is also discussed under the terms *lag* or *temporal feedback compliance* in virtual reality literature, where temporal incompliance or *latency* have been demonstrated to degrade user performance (Bowman et al., 2004).

Karrer (2013) criticises collapsing the articulatory and semantic distance into one dimension as proposed by Hutchins et al. (and maintained by Frohlich) as overly blunt. For his design space for media navigation interfaces, he proposes to split cognitive directness (distance) into these two components. The *semantic distance* is then the complexity of transferring from a conceptual task model to the target domain and the objects involved in representing it, while the renamed *syntactic distance* describes the complexity of transforming between the target domain objects of interest and their representation in the interface. In his work, Karrer shows that many types of interfaces for media navigation minimise one of these two distances but neglect the other. For instance, the dominant video navigation interface, the timeline slider, has a high semantic distance for content-centric navigation tasks such as finding a specific shot or scene in a video, since the user must know or discover on-the-fly "the semantic mapping from the video's semantic structure into its syntactic structure to successfully navigate the video" (Karrer, 2013).

Zeltzer (1985) discusses the continuum from direct to abstract interactions in computer animation. He structures character animation methods by abstraction, from the *guiding* systems that use little to no abstraction, such as manual pose specification or computer puppetry, to *animator-level* and *task level* systems that provide more abstract, high-level control via scripting motions or describing motion goals, respectively. Zeltzer's taxonomy relates to the engagement dimension, with guiding level systems requiring manipulation interfaces and animator- and task-level systems more related to conversational interfaces.

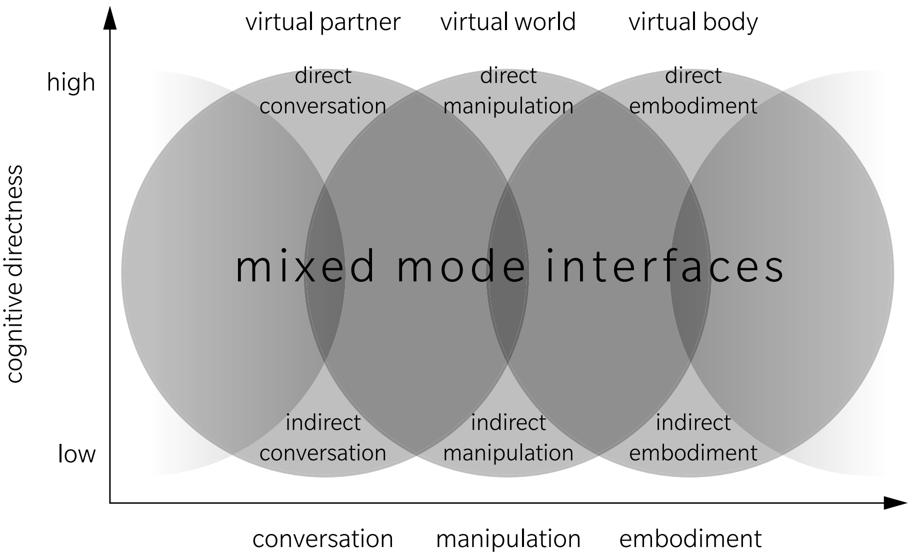


Figure 3.2: The model of cognitive directness and engagement in major mode of interaction metaphors originally proposed by Hutchins et al. (1985) and streamlined by Frohlich (1997) can be extended by *embodiment* as ultimate engagement with the target domain.

3.1.3 Integrality

Spatial editing tasks involve changing spatial parameters—the DOF—of virtual objects. It is often desirable to control many of these in a coordinated fashion. Integrality characterises DOF coordination in input devices.

Theory on the perceptual structure of visual information (Garner, 1974) states that some visual parameters, such as object translation and orientation, or lightness and colour are perceived integrally as a sensory unit, whereas others are perceived separately, such as colour and translation. Jacob et al. (1994) apply this observation to human-computer interaction by pointing out that manipulation is the changing of visual parameters. They characterise *integrated* control as the ability to change a device's input dimensions at the same time, i.e. move across the input space with a Euclidean metric, as opposed to *separated* control, where only one input dimension can be changed at a time, i.e. move across the input space with a city block metric. They propose a measure for a device's integrality in matching tasks as the ratio of Euclidean to rectilinear (Manhattan or city block) distance, a larger value indicating higher integration of control dimensions. Through experiments they find that performance improves when the structure of the perceptual space of a graphical interaction task mirrors that of the control space of the input device.

Next to Jacob et al. several other researchers have characterised and measured coordination of several DOF in spatial manipulation tasks. Zhai and Milgram (1998) observe that the measure of Jacob et al. does not consider the magnitude of how each input dimension contributes to integral control, and propose the measure of *efficiency* that takes this into account. Balakrishnan and Hinckley (2000) propose the measure of *parallelism* in symmetric bimanual control in an object tracking task. The *degree of integration* used by Beaudouin-Lafon (2000) is a measure of how well input device and task DOF match. Kipp and Nguyen (2010) find that most coordination measures in the literature are not suitable for evaluating coordination in free tasks since they assume a target, such as in selection tasks. To solve this they devise an *overall coordination measure* that can be calculated independently of any target.

Computer animation often involves domain objects with incredibly large amounts of degrees of freedom: even a simple 3D articulated biped will have around 30 DOF. This challenge in controlling high numbers of DOF has been termed the *degrees of freedom problem* (Zeltzer, 1985). Approaches to solving this are to use specialised high-DOF input devices, artificial separation or by constraining mappings based on a certain model (see chapter 2).

3.1.4 Correspondence

The concept of kinaesthetic *correspondence* describes the similarity of the physical input motion and the resulting system response (Hinckley and Wigdor, 2012). It has also been referred to as *spatial feedback compliance* (Bowman et al., 2004) or *compatibility* (Beaudouin-Lafon, 2000). Given time, users can adapt to non-correspondences to a certain degree (Cunningham and Welch, 1994). For instance, in typical desktop interaction, in order to move a screen cursor up and down, the mouse is moved forward and backward.

Bodenheimer et al. (1997) distinguish performance animation controls by degree of abstraction in the sense of correspondence. At the one end of the spectrum, mappings are primarily concerned with the character or style of the motion rather than literal mappings between performer and target. Such mappings are more commonly used in computer puppetry. At the other end of the spectrum are efforts to accurately represent motion that strive to limit the degree of abstraction to a minimum (see section 2.3). This relates strongly with the three classes of performance control mappings identified by Neff et al. (2007): *spatial* mappings with a high correspondence between input and result, *spatially based* mappings with a certain degree of correspondence and *abstract* mappings with arbitrary relation between input and output.

3.1.5 Metaphor

Metaphors ground interactions in a familiar framework of concepts that are already understood. They help to establish user expectations and encourage predictions about system behaviour, greatly aiding the ease of learning and use of an interface. Metaphors in human-computer interaction exist at different levels. A higher level at which to structure them are *mode of interaction metaphors* or interaction models, which organise how the user thinks about interacting with the computer (Neale and Carroll, 1997). In the *conversation* metaphor the user engages in a written or spoken dialogue with the machine. In the *manipulation* metaphor, the user acts upon a virtual world rather than using language as an intermediary. In *embodiment*, a user identifies with parts of a virtual world in a more literal way, becoming the target rather than just manipulating it. In the following, each is briefly discussed along with common *task domain metaphors* (Neale and Carroll, 1997) underlying UIs for spatiotemporal media.

Conversation

The conversation metaphor underlies interfaces that use language for communicating between human and machine. These can range from natural to abstract or symbolic languages. Task domain sub-metaphors are scripting environments, command line interfaces, menus, forms, and dialogue boxes. Language-based schemes can also be employed for representations of time-dependent multimedia. These are generally formulated on parallel and sequential programming language features, and include the concept of scripts derived from theatrical works (Little, 1994). In computer animation, language plays an important role in defining behaviour sequences and motion patterns (task-level and animator-level systems in the categories of Zeltzer, 1985). Command lines and scripting environments are a regular feature of mixed-mode interfaces in modern motion design software.

Manipulation

The manipulation metaphor became popular in the 1980s as an alternative to conversation interfaces. It is characterised by the four principles *coherent real-world metaphor*, *natural actions*, *continuous representation* and *responsive visualisation* (Frohlich, 1997) and has been attested a number of usability benefits, including learnability, enhanced expert performance, memorability, fewer errors, better feedback, reduced anxiety and increased control (Shneiderman, 1997). Literature differentiates between *direct manipulation*, in which the user directly engages with the object of interest, and *instrumental manipulation* in which the user engages through some intermediary (cf. Beaudouin-Lafon, 2000).

Direct manipulation is a universal metaphor common to spatial UIs. Direct manipulation implementations can be differentiated by sub-metaphors (e.g. "sticky" or "magnetic" fingers) or by the mechanics of relating input to target, *kinematic* or *physics-based* control. Kinematic controls are more common and offer predictable behaviour, while physics simulations offer more realistic target behaviour and more possibilities for interaction (Wilson et al., 2008).

In frame-based animation, scene features are kinematically manipulated. Moscovich et al. (2005) demonstrated the potential of interactive surfaces for direct manipulation of digital puppets.

Instrumental manipulation is a broad category encompassing all kinds of handles, widgets, gizmos and tools that are used to operate on a target. Such instruments are usually based on task domain sub-metaphors. Examples given by Beaudouin-Lafon (2000) are menus and toolbars in WIMP interfaces. Handles or widgets are instruments employed when device input DOF and object DOF do not match, and are common in desktop 3D editing (cf. section 2.1). In personal mobility, the operation of devices for transportation is also achieved through the use of instruments, and the *steering* metaphor is typical for travel in 3D environments too (Bowman et al., 2004). Computer animation makes heavy use of specialised instruments. In frame-based methods, path descriptions are spatial abstractions of motion that can be used for motion editing. Temporal navigation and transformation metaphors commonly employ the *time plot* metaphor, which also enables indirect spatial manipulation through function graphs of spatial parameters, and *triggering* is used for playback controls in almost all multimedia UIs. Performance controls can also make use of instruments, such as hand-held 6DOF input devices (e.g. Dontcheva et al., 2003), pens (e.g. Oshita, 2005) or digital puppetry contraptions (Sturman, 1998; Jurgensen, 2008).

Time plots are graphical depictions of time. It is widely understood that humans reason, imagine and communicate about the abstract concept of time in a more concrete domain of physical experience, three-dimensional space and motion. The "time is space" metaphor has been thoroughly studied in linguistics (see Evans, 2004, for a listing of literature on this subject), gestural communication (e.g. Cooperrider and Núñez, 2007), and literary studies (Mitchell, 1980), and evidence in the cognitive sciences amounts that the occurrence of this metaphor in communication attributes to a fundamental conceptualisation of time in terms of space (Casasanto and Boroditsky, 2008). Graphical depictions of time are almost as old as the written word (Grafton and Rosenberg, 2010). They are used for organisation in everyday life, such as timelines, calendars, time charts and timetables (Tufte, 1990), and can have such specific uses as presenting patient history in radiology (Dionisio and Cárdenas, 1998). Graph representations of time-dependent media such as timelines, flowgraphs, timed petri nets and temporal hierarchies aid in visualising synchronisation semantics in multimedia systems (Little, 1994). The timeline is possibly the most well-known metaphor for graphically depicting time, and a significant amount of research has been conducted on extending this linear scan metaphor, for instance with content-based markup (e.g. Hürst et al., 2004; Brachmann and Malaka, 2009; Pongnumkul et al., 2010). The two most common time plots used in frame-based animation are the one-dimensional timeline for viewing and editing keyframes and the two-dimensional function graph that plots a single (usually spatial) parameter as a function of time.

Path descriptions are global descriptions of motion (Baecker, 1969). They can take the form of precise trajectories of individual features or generalise motion of whole feature sets, such as articulated bodies (Gleicher, 2001). Although Baecker (1969) uses a broader definition of the term, here it is used in a stricter, spatial sense synonymous to *motion paths*. In their simplest form, motion paths can only convey two- or three-dimensional location. Additions such as oriented, temporally spaced "tick" marks along the curve can convey orientation and timing information to a certain degree. Path descriptions are in use in a wide variety of fields, ranging from motion analysis to wayfinding and navigation systems, for instance for sketch-based interaction techniques for navigating virtual worlds (Igarashi et al., 1998; Hagedorn and Döllner, 2008). In computer animation this metaphor is employed to visualise and edit motion (see section 2.2).

Triggering is a metaphor borrowed from engineering in which small immediate input through a triggering device results in immediate reaction. Interfaces can be as simple as a button or key, since input signals need no spatial information. While not the premier metaphor for spatial navigation, state-based navigation can be appropriate with designed technical constraints. An example is grid-based navigation in old video games, where four arrow keys were used for a discrete navigation through a spatial grid. Triggering is a common interface metaphor for switching through discrete temporal states in time-based media. Triggers can either change the state of playback speed (e.g. play, reverse, fast forward, fast reverse), or provide for discrete navigation in constant fixed time steps (e.g. skip controls for audio/video, turn controls for strategy video games) or pre-designed cues based on content semantics (e.g. DVD chapter access). Trigger input has also been investigated for real time motion editing, as discussed in section 2.3.3.

Embodiment

Interfaces in which the user fully identifies parts or all of his body with a virtual target employ the embodiment metaphor. Here the term embodiment is used in a stricter sense than elsewhere (e.g. Dourish, 2001), as a mode of interaction metaphor in which users cognitively embed themselves in a virtual world.

Physical locomotion interfaces embed the user in the avatar's view for navigating immersive virtual spaces, striving for a maximal likeness of virtual travel to real-world human locomotion (Bowman et al., 2004). Embodiment need not be restricted to literal mappings. For example, in the finger walking metaphor the user embeds his fingers in a set of virtual legs, which has also been proposed for navigating virtual environments (Kim et al., 2008).

Embodiment is common in performance animation. With entirely congruent mappings from performer to character, feedback from the system becomes less important, as the performer can rely on proprioception, training and experience. Partial motion capture, such as either only the body or the face, is not uncommon in performance animation, and real time visual feedback can aid performers to achieve a desired effect (see section 2.3). Less literal mappings fall under embodiment too, such as when puppeteers control a character's mouth with their hand like a sock puppet (Jurgensen, 2008), or use finger walking for motion control (Lam et al., 2004; Lockwood and Singh, 2012).

3.1.6 Manuality

Ever since WIMP was established as the leading form of operating computers, interaction in spatial tasks has been inherently one-handed. Since the 1980s researchers have thus been investigating two-handed manipulation in the human-computer interaction context.

Possibly the most important contribution came from the field of behavioural psychology. In his seminal paper on modelling skilled manual activities of humans, Guiard (1987) identified three classes of human manual activities: *unimanual activities*, such as dart throwing or brushing one's teeth, *bimanual symmetric* activities, where two hands play the same role, either in phase or out of phase, and *bimanual asymmetric* activities, where each hand plays a distinct role in the activity. He observed that by far the most common bimanual activities are of asymmetric nature, meaning that both hands act in concert but each in a distinct way. Guiard postulated three higher order principles of asymmetric bimanual gestures for right-handed individuals.

- The left hand delineates frames relative to which the right hand performs (example: in writing, the left hand holds and guides the paper)
- The left hand operates on a higher temporal and spatial scale (macrometric) than the right hand (micrometric): the right hand operates with significantly higher spatial and temporal frequency, but lower amplitude (example: in writing, the left hand moves the paper much slower and for larger distances than the right, which creates small and frequent strokes)
- The left hand takes precedence in action before the right hand (example: in writing, the left hand first positions the paper, then the right hand starts writing)

Guiard's main contribution is the kinematic chain model for bimanual asymmetric activity. It conforms to the three principles by describing both hands as abstract motors in a hierarchical chain, with the left hand more proximate and the right hand more distal.

Guiard's principles have been incorporated in many works in the HCI community, both in 2D (e.g. Buxton and Myers, 1986; Kabbash et al., 1994; Leganchuk et al., 1998) and in 3D interfaces (e.g. Zeleznik et al., 1997; Hinckley et al., 1998; Balakrishnan and Kurtenbach, 1999). The understanding gained in several decades of research is that bimanual control significantly improves both the *directness* and degree of *manipulation*, and that two-handed interaction techniques consistent with Guiard's characteristics make interaction more "natural" (cf. Kabbash et al., 1994; Leganchuk et al., 1998). This is due to manual as well as cognitive benefits of two-handed interaction.

Researchers have applied the principles, especially the right-to-left reference, in many prototype studies. *Manual* benefits have been observed in most cases: significant efficiency improvements regarding task completion time were found with the *toolglass* metaphor (Buxton and Myers, 1986), right-handed manipulation relative to a prop held by the left hand (Hinckley et al., 1998) or relative to view control with the left hand (Balakrishnan and Kurtenbach, 1999). Reasons for such improvements that have been identified are the reduced target acquisition time due to dedicating each hand to a subtask (Buxton and Myers, 1986; Balakrishnan and Kurtenbach, 1999), overlap in the performance of two subtasks (Leganchuk et al., 1998), ergonomic benefits of body-relative gestures (Hinckley et al., 1998) and a fundamental difference of cognitive qualities between unimanual and bimanual control.

The *cognitive* benefits lie in two main qualities of two-handed input summarised by Leganchuk et al. (1998) and Hinckley et al. (1998). Firstly, bimanual interaction can change how users think about a task. Separating tasks into subtasks, as is commonly done for low-DOF input devices, significantly changes the nature of a task. Manipulating entire objects with two hands as an integrated chunk has manual and cognitive benefits, since task hierarchies can be reduced to a single transaction (cognitive chuck) and allow for the hierarchical specialisation of the hands. Secondly, two hands can provide more information than one hand alone. Bimanual control provides passive haptic feedback on the relative position of the hands; this body-relative interaction space has been shown to provide better orientation and cognition, even if each hand acts in different coordinate systems (Balakrishnan and Hinckley, 1999).

In digital frame-based animation, hand use is dictated by the omnipresent desktop editing environment of keyboard and pointing device. In this the left hand does not engage in manipulation, rather activating discrete commands and modifier keys on the keyboard, while the right hand engages in both spatial and temporal continuous operations. Performance animation seldom comments on aspects of manuality; while many setups employ both hands, strategies of bimanual behaviour are irrelevant for literal embodied mappings. Only with the advent of interactive surfaces have the possibilities of bimanual manipulation been considered for puppetry, and bimanual control was shown to be either emergent from free multi-touch deformation techniques (Moscovich et al., 2005) or specifically designed into a control interface (Kipp and Nguyen, 2010).

3.1.7 Space-Time

Computer animation data has multiple spatial and one temporal dimension. This requires that interfaces allow the viewing and modelling not only of static spaces but of their dynamics as well. User actions occur over time, so input and output both have a temporal component. User time is generally referred to as *real time*, which is continuous, the data time as *virtual* or stream time, which is discrete (Little, 1994; Dionisio and Cárdenas, 1998).

On the medium side, virtual time is a *presentation dimension* of continuous media and thus a carrier of information in itself (Steinmetz and Nahrstedt, 2002). Along with the spatial dimensions, virtual time is thus an integral presentation dimension of time-based media. The model of Karrer (2013) for digital media navigation tasks reflects this by describing time as the essential support domain in time-based media, which holds the sample domain (e.g. amplitude values for audio). Independent of media type, the response time of the system is regarded as an important temporal aspect of output. Also known as lag, this describes the time delay in real time between a user input and a system response and is one important measure of the directness of an interactive experience (see section 3.1.2).

On the input side, the role of time has been hardly investigated. The temporal sequence of manual actions has been identified as playing an important role in human bimanual behaviour. The principles developed by Guiard reflect this in the precedence of left hand actions and coarser temporal scale at which it moves (see section 3.1.6). The concept that grasps time on the input side best comes from multimedia research. It includes signals from sensors into the definition of continuous medium (Steinmetz and Nahrstedt, 2002): any form of continuous input such as a mouse, 6DOF widget or touchscreen is conceptually the same as a video, animation or simulation, i.e. spatial values changing over time. On the input side real time is thus an integral part of the *control dimensions* of the input device.

Viewing interactions with spatiotemporal data as a relation of user space-time to medium space-time has little precedent in the literature. Little (1994) describes temporal transformation tasks involving the playback of stream data as relating virtual time to real time (see section 3.1.1). Individual cases can be made for certain mappings, e.g. the manipulation of spatial and temporal data through metaphors common to frame-based motion editing, or the synchronisation of user and medium time in performance animation. But more cases exist, and no means of describing various user-to-medium space-time relations in a structured manner are known to the author. To address this, in the following section categories of mapping real to virtual space-time are proposed, which are sorted into a taxonomy of space-time interactions.

3.2 Space-Time Interaction

Humans inhabit a space-time continuum, and all human action always has a temporal dimension. Thus any kind of interaction between a human and a computer to create or edit motion relates the human's space-time to the medium's space-time. This can occur in various manifestations. For instance, the time it takes an animator to edit a key frame hardly figures into the dynamics of the result. Quite contrary to this are motion capture performers, whose timing is transferred directly onto a digital character.

Considering the omnipresence of time-based media and the abundance of interaction styles for dealing with them it is surprising that the literature lacks a structured approach to aspects of space-time. This section establishes categories for the multitude of relations between user actions in space and time and time-dependent visual media. A taxonomy is proposed that sorts interaction techniques based on which components of real and virtual space-time are involved.

3.2.1 Space and Time Domains

Any interaction with time-based media can be characterised as a *control mapping* between continuous *input* medium, defined by the control dimensions of the user device and a continuous *output* medium, defined by the presentation dimensions of the data. Both control and presentation include one or more spatial and one temporal dimension. Control mappings thus define how real space and real time of users and devices edit or change the dataset or the current perspective on it. Rather than looking at individual (spatial) dimensions, this categorisation approach distinguishes only between the space and time *domains*. There are different ways in which control domains control medium domains, and in many cases only one of the two domains is controlling or being controlled.

For this discussion the notion of *spatiotemporal integration and separation* are introduced, recalling the distinction of changing dimensions in a coordinated or separate manner (see section 3.1.3). The output medium's presentation dimensions can be viewed and edited integrally or separately regarding the space and time domain. For instance, while frame-based animation edits poses and the time instants at which they occur separately, performance-based or procedural approaches usually define motion in an integrated fashion, by recording or calculating spatial events in a fixed time grid (the sampling rate). Likewise, control dimensions can be reduced to one of both domains by ignoring either the spatial or temporal component of input. For instance, when editing a character feature at a certain frame, the user's time is not reflected in the interaction. There is no relation between pose editing (real) time and resulting motion (virtual time)—the animator could spend a few seconds or hours on handling a single pose, the resulting motion timing would be the same. On the other extreme, trigger controls reduce the spatial channel to binary states that only define time instants or intervals.

This paints a varied picture of how either or both domains of real space-time can control either or both domains of virtual space-time. Next, a classification scheme is developed to structure this picture by creating categories of control mappings.

3.2.2 A Space-Time Classification Scheme

The space-time domain approach separates the single temporal dimension from the spatial dimensions, bisecting both control and presentation dimensions into space and time each. Four basic *space-time categories* of mappings can be constructed from the possible combinations of the two sets (control space, control time) and (presentation space, presentation time):

- space→space
- space→time
- time→space
- time→time

These *separated* space-time categories detach the space and time domains both on control and presentation side. This categorisation can be visualised as a two-by-two matrix with the space and time control domains as the rows and space and time presentation domains as columns.

Yet often presentation space and time domains will be modified in an integrated fashion, or spatial and temporal control domains will both figure into the input-output relation. For this two *control-integrated* space-time categories are introduced that cover input-output mappings in which both control domains contribute to the relation

- space-time→space (space→space, time→space)
- space-time→time (space→time, time→time)

and two *presentation-integrated* space-time categories in which both presentation domains are affected by the interaction.

- space→space-time (space→space, space→time)
- time→space-time (time→space, time→time)

The final cases are the fully *integrated* space-time categories

- space-time→space-time (space→space, time→time)
- space-time→time-space (space→time, time→space)

which reflect that integrated control domains affecting presentation domains in an integrated way can be matched in two ways. These ten space-time categories cover all types of mapping user space-time to medium space-time. The ten categories can be represented by a three by three matrix with each cell denoting a unique combination of the spatio-temporal dimensions of input and how they affect the space and/or time dimensions of the medium (figure 3.3). A single

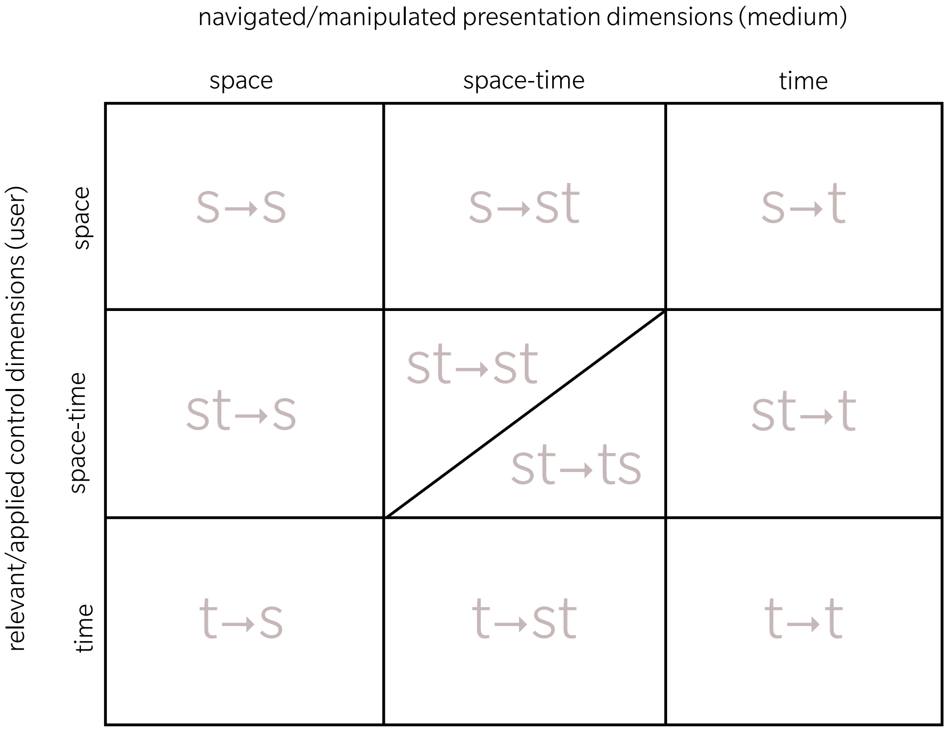


Figure 3.3: The matrix of space-time categories sorts mappings based on how they relate user input in real space-time to medium output in virtual space-time.

cell in the control space-time row, presentation space-time column must then be divided in two in order to reflect the two variants of fully integrated mappings.

An important subset of categories are *symmetric* space-time categories that relate a control domain to its equivalent presentation domain. These are the two separate and one integrated category space→space, time→time and space-time→space-time.

3.2.3 Space-Time Categories

The categories of space-time mappings can be related to the state-of-the-art in interfaces for continuous visual media (see also figure 3.4 for examples).

Space→Space

Controls in the *space→space* category use the spatial component of user actions to affect the spatial dimensions of the medium. Most kinds of interactive editing techniques in computer-aided design fall into this category. In frame-based animation, general (section 2.1) as well as specialised techniques (section 2.2)

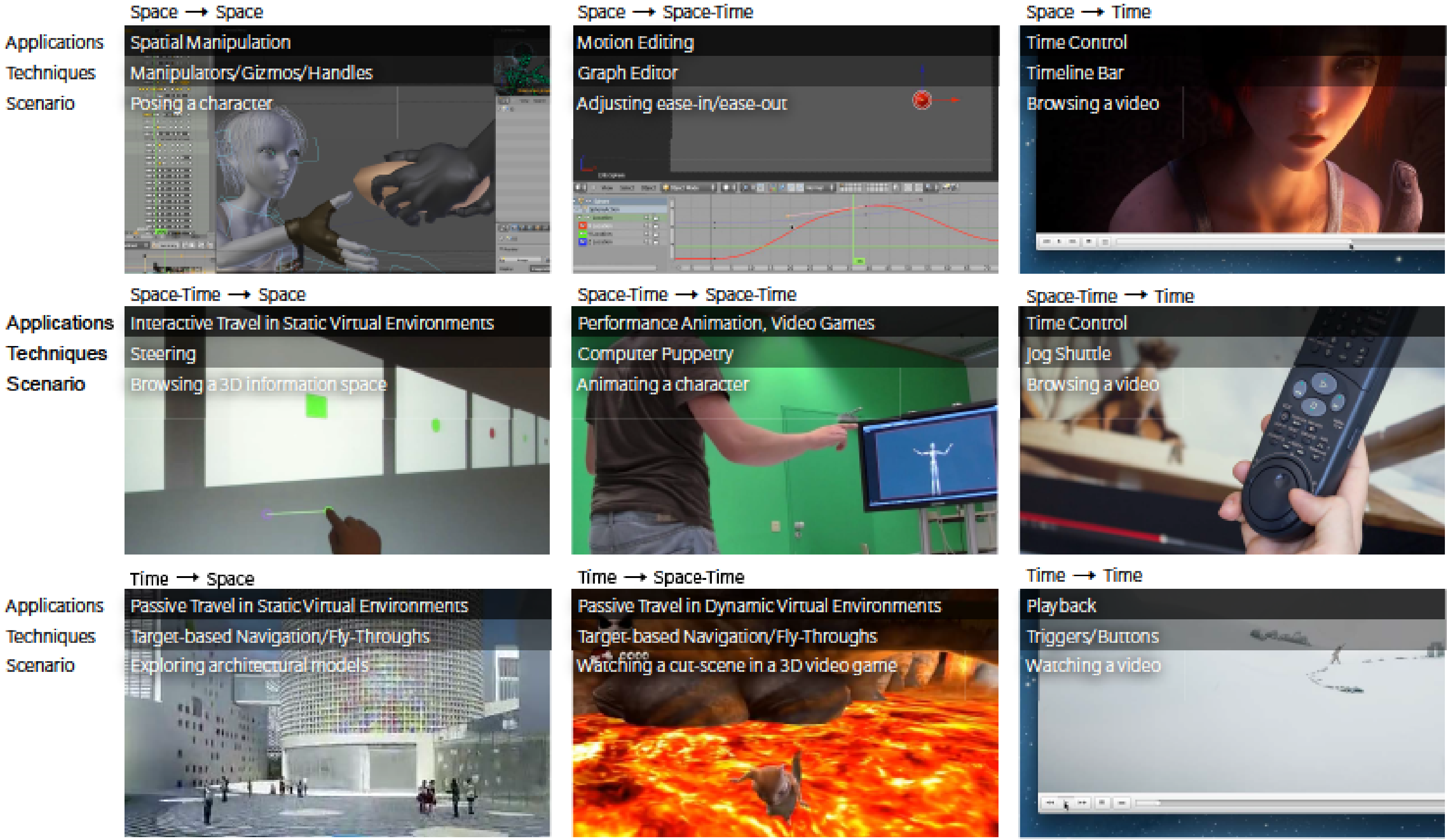


Figure 3.4: Nine categories of space-time mappings with example applications, techniques and scenarios of use. Figure contains cropped stills of third party material licensed under CC BY 3.0. Top left, top right and bottom left images attributed to the Durian Blender Open Movie Project; bottom left image attributed to Frontop Technology Co., Ltd; bottom center image attributed to Apricot Blender Open Game Project.

are used for defining key poses. Edits on a single key frame can propagate over many frames by using interpolation functions. A typical editing metaphor is manipulation.

Space→Time

Techniques in this category employ a spatial representation of time for navigating or altering time-dependent media. This can be abstract time plots or object-based spatial mappings (section 2.2). Software packages for frame-based animation make heavy use of linear time plots for temporal navigation and timing transformations.

Space→Space-Time

In planar mappings from 2DOF input to 2D screen space, time can be represented in one screen dimension, leaving a second screen dimension free, which can be used for representing and altering spatial parameters. Moving points on such two-dimensional time plots can affect both space and time. Two-dimensional time plots are a common means of graphically representing a variable changing over time. Animation packages usually feature a graph editor that enables integrated shifting of key positions and the values they represent in time and one (spatial) dimension.

Time→Space

Some mappings use only the temporal dimension of input to control spatial dimensions of the medium. Since this is just one control dimension, their use is limited. Passive navigation techniques for virtual environments simply map real time to traversal of space (Bowman et al., 2004). After choosing a target or route either automatically or with the user in the loop, the system navigates the user along the route or to the target, mapping user time to medium space. The velocity of the motion is dependent on the technique, and gain ratios can be linear or logarithmic for rapid movement (Mackinlay et al., 1990). This continuous movement over time instead of discrete teleporting increases the spatial orientation of users (Bowman et al., 1997). Editing operations are rare in this category, since the single input DOF is insufficient for most editing tasks.

Time→Time

A straightforward one-to-one mapping of viewer time to medium time is video playback. The spatial component of input requires only a limited number of states, reducing it to non-spatial trigger commands such as play and pause. In the playback state, real time is mapped to virtual time in a linear fashion, usually 1:1. For motion design, trigger-based input has been suggested (Baecker, 1969;

Terra and Metoyer, 2007) and implemented (Zhao and van de Panne, 2005) as a simple means of performance-based timing (see section 2.3).

Time→Space-Time

Sometimes both the temporal and spatial medium parameters depend on the clock input of real time. Automated navigation through a dynamic medium falls into this category. Passive travel techniques need not only relate to static spaces; for time-based media, the techniques used are essentially the same. Scripted camera movement through animated scenes navigates both the time and the space of the target medium. It is often used for cut-scenes in video games, so-called cinematics, when interactive control is taken from the player for a short time in favour of progressing the narrative. This is different from video playback, where the medium is not spatially navigated during playback (videos can spatially only be navigated in the two dimensions of the video frame, e.g. zooming or scrolling the player window).

Space-Time→Space

Certain techniques affect the spatial presentation dimensions based on the spatial and temporal aspects of user input. Velocity-based travel techniques for virtual environments fall into this category. Steering techniques allow the interactive control of velocity and direction, semiautomatic or constrained steering variants take the control of some of these parameters away from the user (Bowman et al., 2004). They place spatial control actions in a time reference; input time determines the exact trajectory taken. Semiautomatic approaches are called for when the designer wants to guide but not completely constrain the user, or the user wants to adjust the travel trajectory. Examples are the target-based *point of interest* technique, which lets a user interactively define a point on an object with a pointer and instantly starts moving the viewpoint toward it at a logarithmic rate (Mackinlay et al., 1990), or the *river analogy*, in which the main trajectory and speed is given and users can change view orientation and steer within the bounds of the prescribed trajectory (Galyean, 1995).

Space-Time→Time

Medium time can also be controlled through continuous spatial input over time. Techniques in this category involve continuous input for controlling playback speed, in which simultaneous adjustment of playback speed via a slider or wheel can change the real time playback, essentially a form of temporal steering. Jog (or shuttle) dials are hardware wheels frequently used in professional video editing either to interactively control the playhead (space→time) or playback speed (space-time→time). In the latter case, spatial input over time determines the exact mapping from virtual to user time.

Space-Time→Space-Time and Space-Time→Time-Space

User interfaces for virtual worlds commonly pair user and medium space-time: spatial actions browse or alter the medium's space, and user and medium time are linearly related. Such mappings are common for interfaces that require high user immersion. Most performance controls for integrated motion creation also fall into this category (section 2.3).

Since a symmetric pairing of the control and presentation domains is more rewarding for most controls, there are no mappings known to the author associated with complement integrated category space-time→time-space.

3.3 Discussion

To conclude, the contributions to a better understanding of UIs for computer animation are summarised. The limitations of the presented design space are debated, and the immanent steps on the research agenda as well as further avenues to explore beyond the scope of this work are outlined.

3.3.1 Contributions

The design space covers seven dimensions characterising important aspects of interfaces for time-based visual media. It is based on design dimensions of user interfaces identified in the HCI literature. It can serve in discussing the issues of current animation tools, and potentially in designing next generation motion design interfaces.

The aspect of space and time could not rely on much prior work, since the role of time in human-computer interaction has only been sparsely treated. The space-time taxonomy can aid describing how user and medium spaces and dynamics relate in human-medium interaction by classifying concepts in viewing and editing controls for videos, animation, simulation, and games based on the involved user and medium dimensions. It can be a valuable framework for analysis and design: by making space and time explicit, researchers and practitioners have new categories to think in.

3.3.2 Known Limitations

The reasoning behind the design space highlighted certain aspects while neglecting others. These limitations concern the design space as well as the space-time taxonomy.

Design Space

Two significant areas of concern for general interface design not explicitly incorporated are feedback and device characteristics.

Responsive, meaningful system *feedback* is a cornerstone of successful interaction (Hinckley and Wigdor, 2012), and literature discusses several dimensions of feedback (Bowman et al., 2004). For motion design interfaces, as with any graphic design tool, the main form of feedback is the state of the target artefact (e.g. object, character, scene), and all current software solutions support interactive preview rendering of the current animation state. Next to this primary feedback certain motion design interfaces require further elements, such as time graphs or motion paths. On a higher level, the interaction metaphor gives guidelines for the feedback, on a lower level it is strongly dependent on technique and device. Strategies for feedback are thus provisionally covered by the design space. Device- and application-specific feedback issues are treated as they arise in the following chapters.

The device-agnostic view from chapter 2 is upheld in the construction of the design space. Including considerations of hardware technology can facilitate but also hamper design reasoning. Instead of including considerations of the design space of input devices (Card et al., 1990, 1991), it maintained more general concepts of directness, correspondence and integrality in spatiotemporal interaction for now. This will ultimately lead to control and display hardware choices in the following chapters.

Space-Time Taxonomy

The examples of time→space, time→space-time and time→time controls include continuous or discrete clock input. These trigger input either at constant clock ticks or through user interaction; in both cases the time of input is relevant. However, when user time is not related to virtual time *and* input has no (or an irrelevant) spatial component, the trigger signals are essentially devoid of space *and* time domain control dimensions. An example are navigation interfaces for discrete, continuous but not time-dependent information spaces (such as image sequences or presentation slides). While these are not covered by the space-time taxonomy, such media are not central to this thesis.

It is important to note that the space-time categories are based on which control domains affect which presentation domains, and ignores other possible dimensions that may be passively involved in the mapping or dependent on the controlled dimension(s) but not directly affected. For example, although typical temporal navigation controls change both the current presentation of space and time (e.g. video frame and timeline playhead), time is the controlled (affected) domain while space is dependent. Mappings can passively involve spatial presentation dimensions, although only presentation timing is affected. For instance, direct manipulation video navigation techniques employ the spatial component of motion for time control (section 2.2). The advantage of a control-centric categorisation is that it is compatible with task-based views that distinguish spatial from temporal editing.

3.3.3 Research Agenda

The design space constitutes a framework with which to discuss motion design tools. Design space analysis can locate an artefact in a design space, but does not offer qualitative assessment. Chapter 4 thus looks at latest-generation interfaces and qualities associated with them in order to inform analysis.

Embodiment is an important concept that should be better established as a mode of interaction metaphor. Extending the directness/engagement model with embodiment next to conversation and manipulation interfaces was a first step in this direction; future work should develop this and relate to other discussions on embodied interaction (e.g. Dourish, 2001).

Chapter 4

Direct Animation User Interfaces

Graphical user interfaces featuring windows, icons, menus and a pointing device—commonly abbreviated to WIMP interfaces—have dominated how we operate computers for decades. Research in disciplines such as virtual and augmented reality or tangible, ubiquitous and mobile computing has explored alternatives to this paradigm, fuelled by advances in computing, sensing and display technology as well as an improved understanding of human cognition. Such UIs offer high fidelity and multimodal sensory output, and recognise expressions of the human body and its surroundings. For lack of a better term they have been classified as post-WIMP interfaces (van Dam, 1997). It is often claimed that these are more natural or intuitive to use than the previous generation of interfaces. Specific benefits attributed are reduced cognitive effort, increased ease of learning and improved task performance.

A trend in research on computer animation interfaces criticises the complex and abstract nature of current motion design tools based on the WIMP paradigm in favour of techniques that are more accommodating to beginners and advanced users alike. These efforts follow the common goal of making animation interfaces easier, more accessible and more supportive of experimentation. Although this trend in computer animation has much in common with what van Dam collects as post-WIMP, it was thus far not linked to that movement.

This chapter connects research on motion design interfaces with the discussion on next generation interfaces in HCI. By locating types of animation interfaces in the design space (chapter 3) as well as in the framework for post-WIMP interfaces by Jacob et al. (2008), one can derive insights on how to design interfaces to feature certain qualities of use. This leads to the concept of *direct animation interfaces* and a set of respective guidelines.

Section 4.1 reviews post-WIMP interfaces, benefits attributed to them, and related conceptual frameworks. Section 4.2 connects this to computer animation with an analysis that culminates in design guidelines for next generation interaction concepts. A discussion of contributions, limitations and future work in section 4.3 concludes the chapter.

4.1 Post-WIMP User Interfaces

Van Dam (1997) lists four main issues of the windows, icons, menus, pointer paradigm:

- The profusion of widgets and features aggregates complexity
- Users spend too much time manipulating the interface, rather than the application
- They were designed for 2D rather than 3D tasks
- Limitations of mouse and keyboard lead to frustration and ergonomic issues

Van Dam argues for a new generation of interfaces that address these issues by offering ways of interacting with computer systems through 3D input devices, gesture recognition, and speech instead of menus, forms, and toolbars. He terms these post-WIMP user interfaces, which he defines as interfaces"containing at least one interaction technique not dependent on classical 2D widgets such as menus and icons". These interfaces commonly (but not necessarily) employ more than one type of communication channel or modality for input and output, such as gesture and voice.

The last two decades have brought post-WIMP interfaces closer to everyone, with new motion-based input for games and simulations, multi-touch screens or gesture controls for mobile devices, to name but a few. Post-WIMP styles of interaction are often described as more direct, intuitive, or natural. Yet it is not entirely clear what this means regarding the interaction experience, and terms are used indiscriminately.

In order to better grasp the unique qualities of post-WIMP interfaces, section 4.1.1 reviews the terms most often used in literature. Reality-based interaction by Jacob et al. (2008) is currently the most comprehensive conceptual framework for post-WIMP user interfaces. It is summarised in section 4.1.2.

4.1.1 Related Qualities

The term *direct* has been used to describe interaction ever since the first manipulation interfaces appeared. The attributes *natural* and *intuitive* are also highly common in describing desirable interfaces (or their use), and advertisement campaigns have adopted these terms and brought them into public discourse.

Direct

A direct interface minimises cognitive and physical indirection between user and application. Direct manipulation was for a long time synonymous to the WIMP paradigm (Shneiderman, 1982). This view has been differentiated since, in that directness is a universal quality of various modes of interaction (Frohlich, 1997), and manipulation interfaces are often only direct for manipulating tools (van Dam, 1997; Beaudouin-Lafon, 2000). Advances in interactive surfaces have brought new topicality to the discussion on directness, as these unify input and display surface, typically localising feedback on user input to the physical points of contact (Hinckley and Wigdor, 2012).

The term is somewhat overused, as it has been employed in various contexts with different, if overlapping, meanings. It can be used both in a concrete physical as well as a more abstract cognitive sense (section 3.1.2). It is nevertheless an important aspect to consider, which is why it is included in the design space of animation interfaces.

Natural

The adjective "natural" is often used for describing our interactions with a system or the interface itself. According to Norman (2002), natural mappings take advantage of physical analogies and cultural standards and lead to immediate understanding. Sturman uses naturalness to describe a subjective evaluation of interaction, that by dictionary definition means free "from artificiality, affectation, or constraint" and "obviously suitable for a specific purpose". He identifies four aspects of naturalness regarding human whole-hand input: using pre-acquired sensorimotor skills, existing hand signs, no intermediary devices, and kinematic mappings that relate task DOF well to the hand DOF (Sturman, 1991). A general definition for natural user interfaces is given by Hinckley and Wigdor (2012): the experience of using a system matches user expectations, it is always clear to the user how to proceed, and only a few steps are required to complete common tasks. Wigdor and Wixon (2011) see the term natural as a design philosophy to create a product, which should mirror the capabilities of the user, meet their needs, take full advantage of their capacities, and fit their task and context demands. It should take full advantage of the user's bandwidth and work like an appendage, an extension of their body.

Intuitive

The adjective intuitive pops up all over research on latest-generation interfaces. Sturman (1991) remarks that natural actions also tend to be intuitive or ingrained behaviours, whereas Wigdor and Wixon (2011) stress that "natural does not mean primitive, or even intuitive". Naumann et al. (2007) point out that intuitive use can only be attributed to the human-machine interaction in a certain context, for the achievement of certain objectives, rather than to a technical system per se. They define intuitive use of a technological system as the user being able to interact effectively, not-consciously, using previous knowledge, in the context of a certain task.

4.1.2 Reality-based Interaction

The common denominator of the above definitions is the role of the user's prior knowledge of concepts from the real world. This led Jacob et al. (2008) to coin the term *reality-based interaction* (RBI) for discussing post-WIMP interfaces. They observed that interaction styles such as virtual, mixed and augmented reality, tangible interaction, and ubiquitous and pervasive computing all build on the user's existing knowledge of the everyday, non-digital world to a larger extent than previous generations of interfaces. They present a framework that can be used to understand, compare, and relate these types of interfaces. It identifies four themes (figure 4.1):

Naïve physics (NP) Every human has an innate understanding of basic concepts of physics, such as gravity, forces and friction.

Body awareness and skills (BAS) Humans have a good sense of their body's posture and capabilities (kinaesthetic and proprioceptive feedback) and the spatial relation of their body's features.

Environment awareness and skills (EAS) Humans perceive and mentally model their environment and place themselves in relation to it.

Social awareness and skills (SAS) Humans are social animals, and generate meaning by relating to other human beings.

These themes cover the main characteristics of post-WIMP interfaces, and can replace, or at least complement, prior discussions on direct, intuitive or natural UIs. The framework can be used to better assess which aspects of the real world can and should be used when designing modern human-machine interfaces. Jacob et al. consider the trend towards reality-based interaction to be positive, since it can *reduce mental effort* to operate a system, *increase learnability*, *improve performance* and *encourage improvisation and exploration*. The authors further note that simply designing a user interface as close to reality is not sufficient, since there are many other qualities to consider: *expressive power*, *efficiency*, *versatility*, *ergonomics*, *accessibility* and *practicality*. They highlight the design tradeoffs between the four RBI themes and these six qualities.

4.2 Post-WIMP Computer Animation

The trend to more direct and natural use can also be observed in research on UIs for computer animation, but has not yet been analysed in a structured manner.

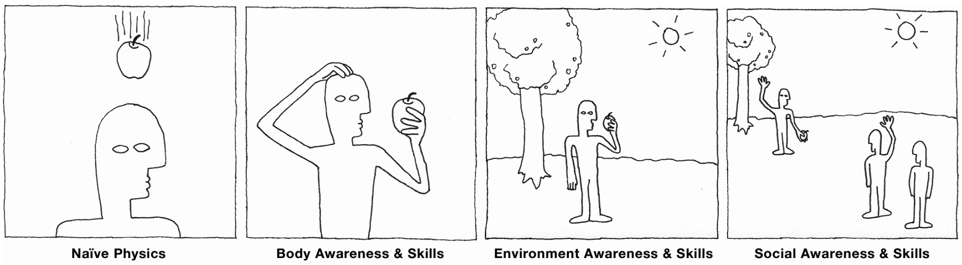


Figure 4.1: The four themes of the reality-based interaction framework for post-WIMP interfaces (images taken from Jacob et al., 2008)

In the following, post-WIMP trends in computer animation are approached with the design space of chapter 3 and the RBI framework. Based on this, the implications for how future computer animation user interfaces can be made to be as direct, natural, and intuitive to use as possible are considered.

4.2.1 Structured Analyses

The design space of animation interfaces enables assessing post-WIMP qualities in computer animation more specifically regarding seven design dimensions. While the RBI framework does not offer specific design dimensions, identifying RBI themes in computer animation tools can additionally inform analysis.

Computer Animation Interfaces in the Design Space

The three computer animation methods or disciplines can be placed in the design space of animation user interfaces (chapter 3). Figure 4.2 shows an assessment of the four ordinal measures as well as a categorisation by the three nominal dimensions. This allows us to further observe the interaction paradigm underlying digital tools most commonly used in each discipline. In algorithmic or procedural animation motions are defined implicitly by abstract representations usually employing some kind of language. Keyframe animation is highly reliant on editor windows, special menu commands and instruments requiring pointer-based manipulation. Performance animation uses high-fidelity input from hand to full body motion, often combined with immersive feedback and is thus most aligned with post-WIMP. Also, factors such as directness and correspondence increase from pre- to post-WIMP animation styles.

RBI Themes in Computer Animation Interfaces

Jacob et al. (2008) illustrate their four design themes with examples and quotes from contemporary research in HCI. The following does the same for computer animation publications. Unfortunately, few papers really discuss the artist's needs and skills. Many arguments are taken for granted or are implied, so that only works can be quoted that make these considerations explicit.

Many techniques aim to tap the artist's innate understanding of space-time processes, relating to the theme of naïve physics (NP). The environment awareness and skills theme (EAS) comes into play as soon as humans interact with these real world space-time processes. Techniques for interactive editing of physical motion simulation by Laszlo et al. (2000) "allow human intuition about motions to be exploited in the interactive creation of novel motions" (NP). The multi-finger deformation technique for 2D puppetry on interactive surfaces of Moscovich et al. (2005) can "make it easy for people to create simple 2D animations by relying on their natural sense of timing, and on their experience with real-world flexible objects" (NP, EAS). In fact, any technique based on motion capture for defining dynamics "relies on users' intuitive sense of space and time" (Davis et al., 2008) (NP, EAS).

Digital puppetry controls use the performer's understanding of his body—the RBI theme body awareness and skills (BAS). As Kipp and Nguyen (2010) illustrate, a "puppeteer uses complex coordinated hand movements to bring a wooden puppet to life. Everybody uses equally complex hand movements to master simple everyday tasks like tying ones shoes" (BAS, EAS). Even the technique for low-fidelity input via mouse and keyboard of Laszlo et al. (2000) "exploits both an animators motor learning skills and their ability to reason about motion planning" (BAS). This is most evident with literal mappings, since "many people find it fairly intuitive to "perform" or mimic the desired motion with their hands" (Terra and Metoyer, 2007, quotation marks taken from source) (BAS), but also holds for non-literal mappings, such as "the ability of human fingers on simulating the motion of legs" (Lam et al., 2004) (BAS).

Collaboration in computer animation is common, as large productions require teams to work together. However, the work on individual elements—a character's motion or the exact dynamics of a scene—is typically a single-user affair. Multi-user controls tap the ability of humans to relate to other human beings—the social awareness and skills RBI theme (SAS). As Sturman (1998) points out, character controls separated across performers requires these to "monitor their own actions, while at the same time, work together to create a unified personality." (BAS, SAS). Even experiments with remote collaboration setups show how "actors are able to interact and influence each other in a shared virtual space in real time" (Nguyen et al., 2010) (SAS).

The general benefits attributed to latest generation interaction styles in computer animation echo those cited in the context of RBI: "this is much faster and more intuitive than traditional temporal keyframing, because the user need not mentally translate static keyframes to temporal motion during design" (Igarashi et al., 2005b) (reduced mental effort); "performance animation makes possible a spontaneous and improvisational exploration of ideas as opposed to the traditional pose-to-pose approach which requires planning ahead" (Kipp and Nguyen, 2010) (encourage improvisation and exploration).

	programmed animation	frame animation	performance animation
task versatility	high	medium	low
metaphor	conversation	manipulation	embodiment
manuality	n/a	unimanual	bimanual
directness	low	medium	high
correspondence	n/a	medium	high
integrality	n/a	medium	high
space-time mappings	(none)–>space-time	space–>space space–>time	space–time–>space-time
interface generation	command line	WIMP	post-WIMP

Figure 4.2: Design space analysis of computer animation user interfaces.

4.2.2 Implications for Design

The analysis using the design space and the RBI framework have implications for designing animation user interfaces.

Task

The main tasks in motion design are motion creation and motion editing. For both, a sense of how real-world objects move depending on weight, size and shape is essential (NP). Further required skills vary with the underlying control metaphor (see below). Regarding versatility, a uniform interface following RBI themes is more desirable than a compound interface (e.g. diverse editors), as long as it supports the same range of tasks. *Designers should strive for interfaces to support both motion creation and spatial and temporal editing tasks within a coherent technique or metaphor.*

Directness

Directness is strongly related to the qualities attributed to RBI. For spatial control in manipulation interfaces, direct interaction requires *simple transformation functions and minimal orientation, location and time offset between input and output*. Direct input devices with their input/output congruence are ideally suited to fulfil these requirements. Since graphical representations of time introduce indirection for content-centric timing tasks, *temporal control should also employ spatial mappings* in order to minimise semantic distance (Karrer, 2013). Direct manipulation video navigation controls are formidable examples of using spatial mappings for time control (Kimber et al., 2007; Dragicevic et al., 2008; Karrer et al., 2008; Goldman et al., 2008).

Correspondence

A high spatial correspondence between input and output requires less mental effort since it draws on our experience in using our own body and encountering real-world objects (BAS, EAS). Yet over time, spatial mappings can also be learnt. Spatial UI designers must face the trade-off between better learnability through high correspondence and the range of motions that can be expressed. For motion creation and editing in particular, *a high correspondence of spatial mappings is desirable*, since the animator wants to focus on the style of the motion rather than the controls for creating it, often in real time.

Integrality

Jacob et al. (1994) recommend that the control structure of the input device should match the perceptual structure of the task, e.g. whether controlled DOF are perceived in an integrated or a separated way. Given a certain device input structure, it might be better to match this by changing the task structure (Martinet et al., 2012). The consequence for unconstrained 3D spatial control is that *motion design interfaces should ideally feature 6DOF input devices*. If other considerations lead to *using lower-DOF input devices, tasks should be adapted according to the device structure*, e.g. by separating translation and orientation. Regarding motion design, considerations of spatiotemporal integrality arise. Motion creation tasks can involve integral spatiotemporal control (e.g. through motion capture), while editing tasks often require to separate spatial and temporal refinement and adaptation. Thus *motion design interfaces should support both integral and separate space-time control*.

Metaphor

Conversation interfaces are well suited for high-level operations, but less suited for spatial precision and expression.

Manipulation interfaces tap our naïve understanding of the laws of physics (NP), our motor memories (BAS) and how we perceive and interact with our surroundings (EAS). Manipulation using instruments requires more learning and mental resources, as well as introducing indirection (Hutchins et al., 1985; Zhai, 1995; Frohlich, 1997).

In temporal editing, spatial representations of time create indirection between control and target, as the instrument is inherently offset from the objects of interest (Beaudouin-Lafon, 2000). The input-to-instrument mapping is congruent, but the mapping from input to target motion is arbitrary. Time plots are effective for time-centric operations of the type "move playhead/keyframe to frame x" but less well suited for content-centric time control (cf. Karrer, 2013). This badly matches our qualifications—humans have a good grasp of real world dynamics, but have a hard time formulating this knowledge in time units (cf. Terra and Metoyer, 2004).

Path descriptions as summaries of motion are more proximate to the target, yet interactive creation or editing of motion paths detach input from resulting motion either temporally or spatially. Path drawing techniques require the whole path to be drawn before any system response, since algorithms reason over the entire trajectory (Igarashi et al., 1998; Hagedorn and Döllner, 2008; Popović et al., 2003; Terra and Metoyer, 2004). The motion path is only a mediator for viewing and defining the motion itself, spatially detaching input from resulting motion, which no longer correspond.

Direct manipulation interfaces exploit our knowledge of real-world objects and manipulating them best (NP, EAS) while at the same time offering the highest versatility. In order to reach the true ideal of direct manipulation, interfaces require direct, corresponding mappings but also unified input and output. Rather than making use of metaphors, such direct-input interfaces literally replicate real-world equivalents. The qualities of direct manipulation extend beyond spatial control to temporal control (section 2.2).

An important aspect of manipulation interfaces is whether they employ kinematic or physics-based mappings (cf. section 3.1.5). While RBI themes advise to respect human's ability to reason abut real-world physical processes, it is perhaps precisely our *naïve* understanding of physics (NP) that could make kinematic controls more favourable. Designers should consider the tradeoffs between the predictability of controls and realistic simulation of physics in their applications.

Embodiment is the most engaging mode of interaction metaphor. Literal mappings ensure maximum direction and let the interface fade away. For avatar control, embodied interaction builds on our proprioceptive and kinaesthetic senses (BAS), and can aid our feeling of presence in virtual environments (EAS). However, embodied mappings are limited to literal and few non-literal mappings from human body to virtual world.

While all motion design interfaces benefit from utilising our sense of naive physics, our environment awareness and skills should have priority for manipulation interfaces, while body awareness and skills should be central to embodied interaction controls. The manipulation metaphor provides the best trade-off between engagement and versatility. Thus *interfaces striving for maximum versatility should employ the direct manipulation metaphor.*

Manuality

It is widely accepted that conformity with the established principles of bimanual behaviour in manipulation (Guiard, 1987) significantly improves both the directness and degree of manipulation through manual and cognitive benefits (BAS). Motion design UIs based on manipulation thus have a lot to gain from two-hand input. *Animation interfaces should strive to support bimanual input by using the according devices that track both hands, such as freehand input VR devices or interactive surfaces.*

Space-Time

Casasanto and Boroditsky (2008) note that linguistic and mental representations of duration and displacement are asymmetric: We do not speak and think as much of space in terms of time as we do of time in terms of space. This comes as no surprise, since time only has one DOF, while real world space has three. In the space-time taxonomy, techniques in the categories time→space, time→space-time and time-space→space-time can only control one spatial DOF with real time. The review in section 3.2 illustrates that these are mainly used for passive navigation, rather than for spatial manipulation—any category relating user time to medium space is unsuitable for editing spatial media. Thus *motion creation and spatiotemporal editing techniques should be in the space→space-time and space-time→space-time categories* (integrated spatiotemporal control). *Spatial motion editing techniques should be in the space→space and space-time→space categories* (separate spatial control). Techniques for temporal editing (separate temporal control) can theoretically occur in any category of the third column.

4.2.3 Direct Animation

Post-WIMP trends in computer animation aim to better address the needs of the artist with versatile interfaces, high input bandwidth, and real-world metaphors. This thesis proposes the term *direct animation* for this movement. Ultimately, post-WIMP, reality-based, natural, intuitive or accessible animation would serve equally well. The concept of directness is chosen for the following reasons.

- Direct interfaces are in line with RBI principles. Definitions of directness emphasise real-world metaphors and natural actions.

- The term does not imply literally copying real-world concepts as much as others do, encouraging undogmatic application of RBI themes.

- The recent success of direct input devices, which harmonise with the qualities discussed.

- Despite the increased use of language and embodiment, direct manipulation is still a powerful and versatile mode of interaction, that is based on how we shape the real world around us with our hands. This is especially applicable to animation which is an art form as much as a craft.

4.2.4 User Interface Design Guidelines

The implications for design gathered in section 4.2.2 can now be summarised as design guidelines for direct animation user interfaces.

Task

1) Direct animation interfaces should support both motion creation and motion editing.

2) Interfaces should require as few different techniques as possible, striving for maximum generality.

Directness

3) Spatial mappings should employ simple transformation functions and minimal location, orientation and time offsets between input and output space-time.

4) Direct time controls should abandon abstract in favour of object-based spatial mappings.

Integrality

5) Spatial mappings should utilise 6DOF input devices for integrated location/orientation control; for lower-DOF input devices, tasks should be broken down according to device structure.

6) Direct animation interfaces should provide techniques both for integral and separate space-time control.

Correspondence

7) Direct animation control mappings should strive for maximal correspondence.

Metaphor

8) Manipulation interfaces should draw upon a user's environment awareness and skills, while embodied interaction should consider a user's body awareness and skills.

9) Direct manipulation is a flexible metaphor that can support all motion design tasks if used with appropriate mappings.

Manuality

10) Direct animation interfaces should employ devices that track both hands.

Space-Time

11) Direct spatial control should be in the space-time categories

- space→space
- space-time→space

12) Direct spatiotemporal control should in the space-time categories

- space→space-time
- space-time→space-time.

4.3 Discussion

This chapter analysed current computer animation interfaces and proposed design guidelines for next generation computer animation. To conclude, contributions are pointed out, limitations are discussed, and an agenda for the next two chapters is outlined.

4.3.1 Contributions

The state of the art in computer animation was analysed from two angles. The design space analysis assesses attributes and qualities of animation tools and how they relate to interaction paradigms. It makes existing trends toward post-WIMP animation interfaces explicit and suggests how these can be pursued further. With the RBI framework of Jacob et al. (2008) an instrument was used that was specifically created for post-WIMP interfaces. It was shown that all four RBI trends can be made out in recent motion design interfaces.

The concept of direct animation interfaces was coined for guiding research, development and design in this application area. The notion of direct interaction was justified with its implication in direct-input devices and direct manipulation metaphors as well as its association with RBI principles. Insights from design space and RBI analysis were formulated as design guidelines for direct animation.

4.3.2 Known Limitations

There are two issues with grounding the use of digital systems on the user's prior knowledge of experiencing the real world: lack of real-world analogies and existing virtual-world knowledge. These are not specific to the target matter of this thesis, but apply to RBI in general. However, they warrant discussion as the investigation of direct animation interfaces is affected by them.

Jacob et al. (2008) consider the limits of relying on prior knowledge when a task has no analogy in the real world. Wobbrock et al. (2009) showed with a study of user-defined gestures on multi-touch interactive surfaces that there is no inherently natural set of gestures for performing anything beyond the most commonplace manipulations. The lack of a real-world source domain is a challenge for designers of interactive systems. When designing for functionality that does not have a real world counterpart, it is important to find metaphors that do not mislead users (Neale and Carroll, 1997).

A problem with the ideal "real world" concept is that interactions with virtual worlds are a large part of our everyday experience. The RBI framework assumes interactions with and within the physical world, but disregards that digital interactions are often of a quite different, much more limited nature. People incorporate the knowledge of how they have learnt to operate "traditional" computers into their strategies for using new systems. An example is the phenomenon of *mouse priming*: people are so used to performing spatial tasks in digital environments with one hand that they do not use their off hand on their own accord in systems allowing bimanual control (Terrenghi et al., 2007; Cao et al., 2008; Wobbrock et al., 2009; North et al., 2009). Once instructed about the richer possibilities, most subjects abandon pointer-like interaction and quickly adopt the new techniques. In a study comparing real-world and digital tabletop interaction, Terrenghi et al. (2007) discovered that tasks affording two-handed asymmetrical manipulation in a non-digital version were approached predominantly one-handed and that bimanual input was only used in symmetric strategies. As a solution they suggest interaction techniques that require bimanual control or snapping functionality to compensate for the lack of physical constraints. North et al. (2009) were able to show that real-world versus mouse priming prior to a surface-based spatial task significantly influences task performance, with subjects using a mouse prior to the surface performing significantly worse than those performing a physical task prior to operating the surface. Epps et al. (2006) suspect that mere hints of an underlying WIMP operating system can support user's preference for single-hand, single-finger gestures using multi-touch software.

For digital systems that aid animators in creating the illusion of life the question of real-world precursors is particularly poignant. Essentially, humans can only directly animate one entity—their own body. Yet children "animate" their dolls and toys at an early age. Thus, arguably, most of us have (at least limited) skills in puppetry. It is these real-world skills that are embellished by embodiment and manipulation performance animation interfaces (respectively). Thus even for the artificial procedures of digital animation we can draw on real world knowledge and skills. We know how living creatures look, move, and behave; we can use this knowledge to breathe life into the inanimate. The question of how well we do this is a matter of talent, skill—and the right tools.

4.3.3 Research Agenda

The design guidelines recommend direct input devices with a direct manipulation metaphor for spatial and temporal control. Since 3D direct interfaces are still a long way from becoming reality (Grossman and Wigdor, 2007), planar direct input devices might currently be the best solution. Given these considerations, two deficiencies of state of the art research can be identified. First, there has been little work done on direct input devices for general 3D animation. Prior approaches look only at 2D mappings (Moscovich et al., 2005) or specialised 3D controls (Kipp and Nguyen, 2010). Second, while direct manipulation can provide effective time control for navigation, it remains to be seen whether this is suited for motion timing. It is likely that this approach can address issues of existing performance timing techniques. In chapters 5 and 6 these problems are approached respectively by applying the design guidelines in practice. Chapter 5 documents the design of a direct animation system for interactive surfaces that features motion creation and basic editing via layering. Chapter 6 then focuses on an interactive motion timing technique that uses a direct spatial mapping.

Chapter 5

A System for Direct Motion Creation and Editing

Direct-touch interactive surfaces have become a standard means of interaction in recent years. In the mobile sector they have largely replaced keyboard and stylus and they are increasingly being used in other contexts, such as collaborative work on large screens. Keys to their success are the high input bandwidth and the congruence of input and output space. The potential of interactive surfaces has been explored for various applications, but has only been hinted at for performance animation: Moscovich et al. (2005) demonstrated direct-touch deformation of 2D drawings, and Kipp and Nguyen (2010) developed a surface-based interface for control of a 3D puppet arm. Surface-based 3D control techniques are mostly developed for manipulating objects rather than for continuous motion control. Furthermore, related work mostly illuminates individual techniques, evading challenges that come with a system perspective, like workflow integration and connecting to legacy software.

This chapter investigates interactive surfaces for direct animation. It documents the design and implementation of a multi-touch motion creation and editing system prototype[1]. The design approach follows the direct animation guidelines (chapter 4) and a review of the design space of interactive surfaces. The process involves several challenges, like finding adequate direct-touch controls for surface-based motion capture and layered motion recording. The significant 3D animation functionality required suggests extending an existing animation system. This introduces the further challenge of extending a legacy system not designed for direct-touch bimanual input.

After a review of design knowledge for interactive surfaces in section 5.1, the design rationale is argued in section 5.2, and put into practice in section 5.3. A study evaluating this approach shows that even inexperienced users can successfully use the prototype (section 5.4). A discussion of contributions, limitations and future work in section 5.5 concludes the chapter.

---

[1] Parts of this chapter have been previously published (Walther-Franks et al., 2011b).

5.1 Design Space of Direct-Touch Interfaces

Any design rationale should be based on knowledge gathered in prior research and practice. This section gives an overview of the large body of work on designing direct-touch interfaces. For a consistent discussion, the conceptual framework established in chapter 3 is maintained.

5.1.1 Task

The basic task for many applications is object manipulation involving rigid transformations, i.e. translation and rotation. 2D rigid body transformations using one or more contact points have been well researched (Hancock et al., 2006). Working with threedimensional content poses the challenge of a discrepancy between input space (2D) and output space (3D). In recent years researchers have started investigating 3D manipulation on interactive surfaces, from shallow-depth manipulation (Hancock et al., 2007) over translation only (Martinet et al., 2010a) to full 6DOF control (Hancock et al., 2009; Reisman et al., 2009).

An important secondary task is defining the current perspective on 2D or 3D data. With few exemptions (Edelmann et al., 2009; Fu et al., 2010), research on surface-based 3D interaction has not dealt much with view control. Yet 3D navigation is essential for editing complex scenes in order to acquire multiple perspectives on the target or zoom in on details.

Problems can arise when view and object transformation can phenomenologically not be discerned (this can happen under certain conditions, e.g. a single object in front of a plain background). To illustrate, through-the-lens techniques that calculate transformations in order to maintain input-to-world pick correlation (section 2.1.1) have been used for camera control (Gleicher and Witkin, 1992) and direct object manipulation (Reisman et al., 2009). Mode separation is prevalent in desktop 3D interaction, where virtual buttons, mouse buttons or modifier keys change between object and view control tools. While this is motivated by low-DOF input devices it has the added benefit of making the current mode explicit. Mouse emulation techniques featuring distinction of up to three mouse buttons (Matejka et al., 2009) bring efficient mode control to interactive surfaces. Since the additional degrees of freedom granted by up to five finger touches per hand are needed for button activation, they are not available for further control, reducing a single hand to 2DOF input.

5.1.2 Metaphor

The congruent input and output space of direct input devices promotes a manipulation style of interaction. User studies of surface computing interactions suggests that no “natural” mappings exists beyond the most basic manipulations (Epps et al., 2006; Wobbrock et al., 2009). Most manipulation techniques are kinematic mappings, where individual surface contacts exert a pseudo friction force by "sticking" to objects or "pinning" them down. Kinematic solvers aim to maintain pick correlation between the initial contact and the corresponding "touched" point on the target. As an alternative to this kinematic control, Cao et al. (2008) and Wilson et al. (2008) propose surface-based manipulation through virtual forces. This offers a more comprehensive and realistic simulation of physical forces and is also used in desktop-based and immersive virtual environments (see chapter 2). Comparing their physics-based with kinematic controls, Wilson et al. find that while the former is more predictable than simulated physics, the latter offers more possibilities and emergent styles of interaction. However, the authors also point out that only partially modelling real world forces can cause confusion and frustration as users expect the full quality of real physical interaction in the virtual.

Different metaphors in the same system can enhance the distinction between controls that otherwise have much in common. For instance, in the example of desktop 3D interaction, object editing usually employs the direct or instrumented interaction metaphors, while view controls bear more resemblance to steering. This could also support the mental distinction between phenomenologically similar spatial editing and navigation operations on interactive surfaces.

5.1.3 Integrality

On a device level, planar interactive surfaces with multi-point detection support two degrees of freedom, x and y, per contact. Researchers have proposed manipulation techniques that combine multiple points to create single integrated controls for 2D (Hancock et al., 2006) and 3D (Gleicher and Witkin, 1992; Reisman et al., 2009) rotation and translation. Yet Martinet et al. (2012) point out that for working with 3D data on planar devices, following the dictum of Jacob et al. that the structure of the task should match the structure of the input device is not possible: multi-touch-based surface interaction cannot truly support integrated 6DOF control since human fingers have separable DOF (Ingram et al., 2008). Martinet et al. thus propose the *Depth-Separated Screen-Space* (DS3) technique which allows translation separate from orientation. Like the *Sticky Tools* technique of Hancock et al. (2009), the number of fingers and where they touch the target (direct) or not (indirect) determines the control mode. An experiment on integrated versus separated interaction techniques leads the authors to propose a solution to the dilemma that the criteria of Jacob et al. cannot be fulfilled for such tasks. When faced with a choice, their results suggest that performance is improved when the interaction technique follows the structure of the input device rather than the structure of the task.

5.1.4 Directness

Interactive surfaces can reduce the spatial distance between the user and the target to a minimum. While some studies find little benefits of direct-touch over indirect (mouse) input for precision and accuracy of basic tasks (Sears and Shneiderman, 1991; Forlines et al., 2007), more recent studies attribute direct-touch large benefits over mouse devices (Kin et al., 2009; North et al., 2009). Yet many issues in surface interaction work against maximal directness. The following discusses directness in the context of *fat fingers*, *reach*, *disambiguation*, *dynamic targets*, and *interaction above the surface*.

Fat Fingers

The "fat finger" problem in surface computing is actually two issues—*imprecision* introduced when reducing the finger contact area to a single point, and *occlusion* of on-screen content through the user's fingers, hands and arms (Wigdor and Wixon, 2011). Benko et al. (2006) and Wigdor and Wixon list some approaches that address imprecision, which is mainly a problem for target acquisition. The occlusion issue is important whenever constant feedback on the input is essential, as is the case with most spatial editing tasks. Widgets that additionally display the touched interface element next to the finger contact can solve this issue, and have been successfully implemented for text entry on mobile touchscreens. Yet it is unclear how this technique can be ported to displays of complex, changing data as are common in 3D modelling and animation. Here, re-introducing indirection can alleviate the occlusion problem. With *HybridPointing*, Forlines et al. (2006) demonstrate an interaction technique for direct input on interactive surfaces that enables fast switching between absolute and relative control.

Reach

Absolute input techniques require that the user be able to reach every part of the screen. This may no longer be ergonomic or even practically feasible when the display exceeds a certain size. A solution is to limit the area of interaction to a part of the screen that the user can reach. Again, indirection with clutching mechanisms and input-output gain ratios can help (Forlines et al., 2006).

Disambiguation

The spatial distance between input and target can also be used as a parameter for interaction design. For instance, fingers or pens touching the target can control different DOF than off-target contacts (mode change). Wobbrock et al. (2009) propose to use the target of the gesture for disambiguation in this way. In their example, "splaying 5 fingers outward on an object will *enlarge it*, but doing so in the background will *zoom in*" (emphasis adopted from source). This allows

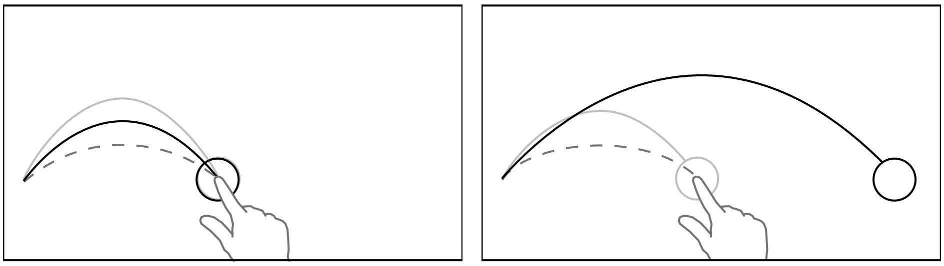

(a) Average blending. (b) Additive blending.

Figure 5.1: Two kinds of blending for layered animation recording. The ghosted trajectory represents the previous animation layer, the dotted line the input motion, the opaque trajectory the resulting motion.

gesture reuse, which is important to increase learnability and memorability (Wu et al., 2006). The *Sticky Tools* (Hancock et al., 2009) and *Depth-Seperated Screen-Space* (Martinet et al., 2012) manipulation techniques also use the spatial offset to discern object rotation from translation. Martinet et al. comment on the problem of this type of differentiation with extreme object sizes or irregular shapes. They suggest dedicated areas for separation (see also Herrlich et al., 2011).

Dynamic Targets

Layered motion recording can involve manipulating moving targets after the initial capture pass. Dontcheva et al. (2003) and Neff et al. (2007) discuss mappings and blending modes in the context of layered motion recording.

An *absolute mapping* assigns to every point in input space exactly one point in output space. This is helpful in establishing basic trajectories or interactions of end-effectors with the environment. *Relative mapping* applies transformation relative to the initial input state. This allows arbitrary input location, and clutching can increase the comfort of use. Both absolute and relative input can be applied locally and globally, which makes a significant difference when controlling behaviour of a feature that inherits motion from its parents. Local mapping allows the user to ignore motion of parent features and concentrate on local transformations. On the downside, this requires the user to mentally perform an inverse transformation of the parent feature trajectory, which can be quite challenging with large rotations.

By default, performance control of a feature overwrites any previous recordings made for it. In this way, performers can practice and test a motion until they get it right. They might however want to keep aspects of an original recording and change others. Blending a performance with a previous recording expands the possibilities for control. It allows performance-based editing of existing animations. Figure 5.1 shows the *average* (Neff et al., 2007) and *additive* (Dontcheva et al., 2003; Neff et al., 2007) blending modes.

Previous research on layered recording for performance animation has only used indirect input devices, such as 6DOF widgets (Balaguer and Gobbetti, 1995; Oore et al., 2002a; Dontcheva et al., 2003) or a mouse (Neff et al., 2007). In these setups indirection is inherent in the spatial offset between input device and screen, making the additional indirection caused when controlling moving targets less challenging. For controlling dynamic content on direct input devices this becomes more of a problem, since the expected pick correlation cannot be maintained in all cases. The same is the case for interactive blending. If output is a combination of current and previous input, this questions the applicability of the pick correlation criterion.

Interaction Above the Surface

Hilliges et al. (2009) describe a vision-based tabletop system and technique for grabbing virtual objects that uses sensed height of users' hands to let them perform a grab gesture and lift a virtual object, essentially allowing 3D translation. This enables true "pick up" and "release" gestures that otherwise are not feasible on touch screens. However, as the authors note, this breaks the close coupling of input and output. To compensate for this loss of directness they developed a shadow-based techniques that projects the users's hand into the virtual 3D scene. The hand's shadow is fused with shadows that virtual objects cast in the scene and provides an additional depth cue. Moreover, it "provides a real-world metaphor to map between actions in the physical space and interactions inside the virtual 3D scene."

5.1.5 Correspondence

Interactive surfaces promote motor and perceptual correspondence between input and output. This correspondence is put to the test when planar input space meets unconstrained three-dimensional virtual space. As Grossman and Wigdor (2007) point out, adaptation of the direct-touch input paradigm to the third dimension is not a simple affair. The fact that the user only directly interacts with the two-dimensional projections of three-dimensional data significantly limits the correspondence between input and output. For instance, to move an object in the screen z dimension, one cannot perform the equivalent motion with standard sensing hardware.

5.1.6 Manuality

Studies by Forlines et al. (2007) and Kin et al. (2009) demonstrated that the benefits of two-handed (symmetric) input also transfer to interactive surfaces for basic selection and dragging tasks. Forlines et al. (2007) even come to the conclusion that direct input significantly outperforms indirect input for bimanual tasks, due to the mental strain of observing two mouse cursors as opposed to using proprioception in surface-based input. The difficulty is to get users to use both hands, since single-handed controls in typical UIs can prime them (see section 4.3.2).

5.1.7 Space-Time

Direct-touch spatial editing is almost exclusively evaluated in the scope of basic object editing in static environments (space→space). Non-spatial trigger input by tapping the screen (time→time) is commonly employed for discrete navigation of image sequences or videos, e.g. TV sports presenters reviewing video recordings of a game. With the exception of Moscovich et al. (2005) and Kipp and Nguyen (2010), the potential of direct touch for motion capture (space-time→space-time) has received little attention in prior research.

Surface-specific techniques thus are mainly aligned along symmetric space-time categories. The absence of passive, time-based mappings or graphical depictions of time might be just because the coupling of input and output so strongly affords direct, continuous manipulation as opposed to tool use or automation. While it is still pure conjecture, it is possible that direct-touch promotes symmetric space-time mappings which couple user and medium space and time more literally, while indirect input might be better suited for more mediated space-time controls.

5.2 Design Approach

A wide-spread model for the interaction design process is human-centred design, in which target users are included from concept to product completion, defining the requirements and giving feedback throughout development. When designing new classes of interfaces that are fundamentally different to operate, this model is less suitable, as target groups find it difficult to think outside known concepts. For this reason, requirements gathering was omitted in favour of an alternative approach. This follows the guidelines for direct animation interfaces (section 4.2.4), refined by design knowledge on spatial control with interactive surfaces (section 5.1) to develop a proof-of-concept direct animation system.

The first decision is to build the system around performance animation, as this method best matches the direct animation guidelines. Section 5.2.1 develops design considerations for the central feature of performance control on direct-touch interactive surfaces. In order to function in complex workflows, general-purpose manipulation techniques must be complemented with unconstrained view controls, for which a design strategy is developed in section 5.2.2.

5.2.1 Performance Control

The problem for surface-based motion capture is to design spatial mappings that allow expressive, direct performance control by taking into account the unique characteristics of multi-touch displays.

Task

Performance control needs to address both intricate tasks such as inclining the head of a character and very general tasks such as unconstrained movement in 3D space. Design guideline 2 recommends a control interface that is able to tackle many diverse tasks rather than being specialised for a specific task, as such specialised tools create additional mental load since each must be individually learnt. While specialised control interfaces use a variety of constraints and mappings to enable specific real time transformation of the target, a more general approach would be to allow unconstrained rigid control, i.e. translation and rotation. For low-DOF input devices, integrated rotation and translation is usually reduced to translation through handles or widgets. Thus translation becomes the main sub-task in a general control interface.

- Performance controls should address the diverse needs of the puppeteer and thus be universal.
- Translation is the most basic control for motion creation; rotation and scaling can be reduced to translation operations.

Metaphor

A general control interface requires a metaphor to match this approach. Design guideline 9 recommends direct manipulation as the most general metaphor for puppet control. Through manipulation the puppeteer can flexibly create and release mappings with a drag-and-drop style of interaction, directness minimises mediation between user and target domain. Regarding kinematic versus physics-based manipulation mappings, realism and emergent control styles stand against precision, predictability and reliability. In animation, full control has a higher priority than realism.

- Direct manipulation best matches the need for general translation manipulation.
- Kinematic mappings ensure the best degree of control.

Integrality

Performance control requires coordination of many DOF. According to design guideline 5, these should be broken down according to device structure if input and target DOF do not match, which on interactive surfaces is essentially only 2DOF control. This may not be as much of a constraint to 3D animation as it seems: since the final rendered result will be a two-dimensional video, motion design cannot fully use the third dimension[2]. Full 3D control can be achieved by additive motion layering: changing the control-display mapping (e.g. by navigating the view) between takes allows control of further target DOF.

- The structure of interactive surfaces dictates 2D control.
- For 3D control the output space can be fully covered by changing control mappings between takes (e.g. rotating the view).

Directness

A close coupling of input and target is desirable for direct animation interfaces (guideline 3). This stands in contrast to the requirement of constant visual feedback, which can be obscured by fingers or pens. Most importantly, directness cannot be maintained with static views on dynamic targets (section 5.1.4). Making view space static to the input reference frame would resolve this issue: By transforming view matrix in the same way that input is transformed, input and output space are again congruent. Another solution is to embrace indirection. Indirect control also solves occlusion problems and enables more ergonomic postures.

- The best of both worlds can be achieved with both direct and indirect control. In this way the choice is with the animator, who can decide on the tradeoff for each individual occasion.
- View attaching enables direct input in dynamic reference frames.

Correspondence

In order to best transfer motion intentions through input movements to animating a digital character, maximal correspondence between input and output is required (guideline 7). The problem with the third dimension on interactive surfaces is that barring above-the-surface input, manipulations in the screen z dimension cannot maintain this correspondence, since input motions can only occur in a plane. True correspondence can thus only be maintained with 2DOF planar input.

- Capture takes place within a plane constraint determined by the target and the current projection.
- Input motion is transferred 1:1 onto the current projection of the scene.

---

[2] 3D displays in cinema and home theatres might just be changing this.

Manuality

To maximise bandwidth, the system should enable both symmetrical and asymmetrical bimanual input (guideline 10). The 2D capture approach implicates that no single spatial manipulation requires more than a single hand. Consequentially, two single-handed operations can easily be combined to enable parallel operation, for instance one hand per character limb, allowing emergent asymmetric and symmetric control (cf. Cutler et al., 1997). The moded operation dictated by indirect control, in which the object selection and translation operations are disconnected, constrains parallel control of individual targets: After selection, the target must be assigned to an input stream until the next target can be selected and manipulated in turn.

- The 2DOF capture approach allows symmetric or asymmetric parallel unimanual control of two targets.

- The only limitation is sequential assignment of inputs to targets, once assigned they can be operated in parallel.

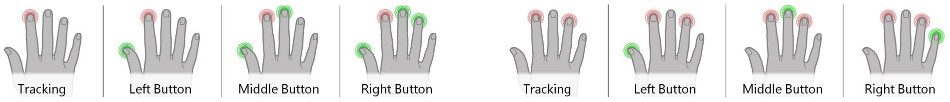
5.2.2 View Control

The design approach for performance control requires means of setting the current view in order to define the control mapping.

Task

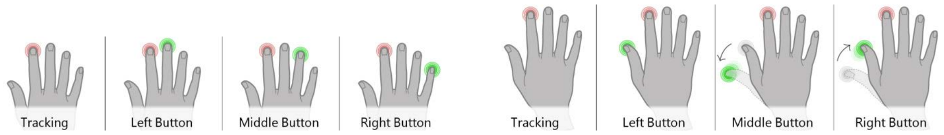
Since performance control mappings are defined by the current projection, this puts a high demand on view controls regarding flexibility, efficiency and precision. Some surface-based virtual reality setups use implicit scene navigation by tracking user head position and orientation. However, this limits the range of control. For unconstrained access to all camera degrees of freedom a manual approach offers the highest degree of control. In desktop 3D CAD applications, free camera controls often split the degrees of freedom into modes based on camera operations in cinematography. Camera "panning" translates in the current view plane, "zooming" along camera z axis and "orbiting" rotates around a centre at fixed z distance from the camera view. While zooming and panning cover the camera's three translational DOF, the third rotational DOF, camera roll, is less essential since the camera up vector usually stays orthogonal to a scene "ground plane". While in desktop environments this DOF separation is mainly owed to low-DOF input devices it can also be employed on devices that allow more integrated transformation techniques, in order to allow more precise control (Nacenta et al., 2009).

- Separated control of camera parameters enable precise view adjustments.



(a) The Chording technique.

(b) The Side technique.



(c) The Distance technique.

(d) The Gesture technique.

Figure 5.2: Techniques for mouse emulation on multi-touch input devices (Matejka et al., 2009) use relative position, number, or movement of fingers to determine left/middle/right mouse button down states (images taken from source).

Integrality

Two important factors for efficiency are easy switching between capture and view operations and dedicating hands to tasks. This requires that a single hand be able to activate different input modes with as little effort as possible. Widgets as the obvious solution produce clutter and interfere with the spatial control tasks. Modal distinction by on- or off-target hit testing is problematic if the target has unusual shape or dimensions and also conflicts with disjoint selection and manipulation for indirect input. A third solution is to use the bandwidth offered by whole-hand input. Matejka et al. (2009) propose four techniques that offer single-hand 2DOF control in various states by using multiple fingers of the same hand for mode distinction (figure 5.2). The *Side* technique requires a single finger for continuous tracking, which was already allocated to performance control. The *Distance* technique does not seem robust to variations in hand size finger posture, and mode switching via *Gesture* is not suitable for continuous spatial input. Multi-finger *Chording* allows robust state distinctions without dependency on a tracking state or gestures but has a higher footprint due to additional fingers.

- In order to separate between capture and view control, multi-finger chording is employed in which the number of fingers switch between modes.

Manuality

If individual sets of camera parameters are controlled with a single hand, this allows emergent styles of interaction. Combining two different camera operations, one with each hand, allows asymmetric view control. For instance, left hand zooming and right hand panning can be combined to simultaneous 3DOF control using both hands. In combination with performance controls, this enables interaction styles that follow principles of asymmetric bimanual behaviour (Guiard, 1987): the left hand can operate view, which will be at a lower spatial and temporal frequency and with precedence to the right hand, which acts in the reference frame provided by the left. This approach can be used to simplify view attaching for editing in dynamic reference frames: attaching the camera to the current reference frame for all camera operations provides the benefits of kinaesthetic reference frames and solves the issue of direct-touch and dynamic targets.

- Unimanual controls allow integrated bimanual view control.
- Two-handed control of view and space supports the principles of asymmetric bimanual behaviour.
- Camera attachment addresses the problem of direct manipulation in dynamic reference frames.

5.3 Prototype System

The design approach was put to the test in the development of a working prototype. Section 5.3.1 discusses the reasons behind the decision to extend existing software rather than building from scratch, which required altering its software architecture (section 5.3.2). Section 5.3.3 describes how basic controls were realised.

5.3.1 Extending Legacy Systems

In exploring new types of animation interfaces, one can either write new software from scratch or build on existing solutions. HCI researchers should be sceptical to adapt legacy systems to new approaches for several reasons.

- *Incompatible metaphors.* New devices and styles require a rethinking of all aspects of interaction. Having to deal with existing concepts can limit thinking “outside of the box”.
- *Incompatible software architecture.* Legacy software is designed around and optimised for a specific interaction paradigm, which can make it incompatible to different paradigms. For example, WIMP software architecture is typically developed for single-pointer sequential control and does not support parallel operation as allowed by multi-pointer input.
- *Incompatible graphical interface.* Interaction styles also define the GUI. For instance, in mouse-based interfaces very precise pointing allows small controls that do not hold up to finger input, which is much less precise.

There are also good reasons in favour of developing a proof-of-concept prototype on the basis of an existing system.

- *Existing functionality.* Building a system from scratch is a daunting task, since professional software includes complex functionality created by large developer teams over years. The scope of this work does not make it possible to develop the software components required for a functioning animation system, leaving the choice between investigating single aspects of a system or modifying an existing one.

- *Integration with other methods.* It is unlikely that the introduction of a new interaction device and/or paradigm will change all aspects of existing practice. It is more likely to be incorporated into an existing workflow. This also requires thinking about compatibility between new and existing paradigms and methods on a system level.

- *Real world challenge.* HCI research is often concerned with novel interaction ideas and visions, neglecting real world problems of how to transform the latest concepts and insights into actual solutions. To these belong questions like "how to innovate interaction when users have heavily invested in tools based on older paradigms?", as is the case for computer animation. Showing how to adapt software already in use to support novel means of interaction is a contribution along these lines.

- *Increased impact.* Research prototypes often only run under certain conditions of use and on specific hardware, since software robustness and portability are of subordinate concern to the issues under investigation. Extending an established software system makes the result more feature complete and usable, which in turn can result in more potential users and a better spread of ideas. Considering the existing user base this can create a much higher level of impact.

In the context of developing a surface-based direct animation system, the arguments in favour of extending existing software outweigh those against this approach. A suitable candidate for extension is the open source 3D package Blender[3]. It features a powerful and complete modelling and animation tool chain, freely accessible code, and there is a large community of Blender developers and artists.

5.3.2 Changes to the UI Software Architecture

Since Blender neither supports multi-touch input nor concurrent operations, changes were necessary to its user interface module, especially the event system. Adaptation was performed at three levels, corresponding to three stages of

---

[3] `http://www.blender.org/`, last accessed 6th January 2019

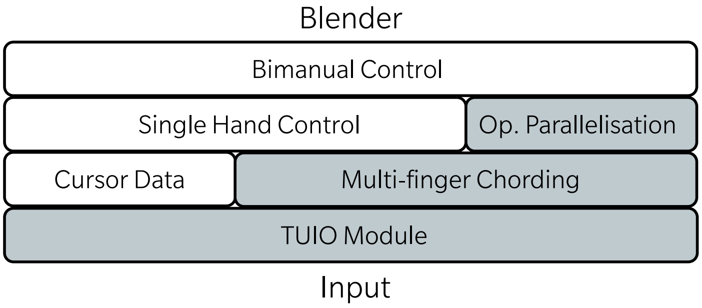


Figure 5.3: Each stage of multi-touch integration is built upon alterations in Blender's event architecture.

multi-touch integration (figure 5.3). The first step established the interface to the multi-touch sensor, internally exposing the tracking data. Multi-finger gestures were mapped to standard input events in the second step, enabling single-handed touch input. In step three, changes to event handling were made so that parallel operations are possible in order to support two-handed input. The following explains the challenges involved and solutions found for these three levels of integration, including details of the implementation.

Input Interface

Multi-touch is a fairly young technology still lacking in standardisation, and a multitude of configurations for sensors, tracking software and operating system support are available. TUIO was chosen as a tracker-client communication layer. It is an open, platform independent framework that defines a common protocol and API for tangible interfaces and multi-touch surfaces (Kaltenbrunner et al., 2005). It is based on the Open Sound Control (OSC) protocol, an emerging standard for interactive environments. The TUIO 1.1 protocol specifies message formats for 2D, 2.5D and 3D surface-based point, blob and marker input. The simplest form a finger blob can take is a 2D cursor, which sufficed for prototype purposes:

```
/tuio/2Dcur alive s_id0 ... s_idN
/tuio/2Dcur set s_id x_pos y_pos x_vel y_vel m_accel
/tuio/2Dcur fseq f_id
```

A c++ implementation of a TUIO client[4] was integrated as a module into the Blender software architecture.

---

[4] `http://www.tuio.org/?software`, last accessed 6th January 2019

Multi-finger Gestures

With the 2D cursors objects available, the next problem was how to connect this to an event system designed for mouse and keyboard. The chording technique for mouse emulation provided the solution, by mapping multiple finger cursors to single 2-DOF input events (cf. section 5.2.2).

The chording technique of Matejka et al. (2009) aggregates contacts within a certain timeout (150 ms). For emulating a single pointer, further input occurring during an ongoing, already registered multi-finger continuous gesture is simply ignored. While this suffices for single-hand input, bimanual interaction requires the recognition of further chorded gestures in parallel. This is achieved by clustering the contacts using a spatial as well as a temporal threshold. Fingers are only added to the gesture if they are within a certain distance of the centroid of the gesture's cursor cluster (figure 5.4), otherwise they create a new multi-finger gesture. After initial *registration* the gesture can be *relaxed*, i.e. the finger constellation required for detection need not be maintained during the rest of the continuous gesture (Wu et al., 2006). This means that adding or removing a finger to the cluster will not change the gesture, making continuous gestures resistant to tracking interruptions or touch pressure relaxation. To reduce errors caused by false detections or users accidentally touching the surface with more fingers than intended, touches are also filtered by a minimum lifetime threshold, currently set to 80 ms.

Each multi-finger gesture has a unique ID assigned to it, which it keeps until destruction when the last of its cursors is removed. The TUIO module handles the clustering and issues internal TUIO events:

```
TUIO_EVENT_ONEFINGERDOWN
TUIO_EVENT_ONEFINGERUP
...
TUIO_EVENT_FIVEFINGERDOWN
TUIO_EVENT_FIVEFINGERUP
TUIO_EVENT_MOVE
```

All TUIO events issuing from the same gesture have the same ID. Blender's main event loop was modified to query the TUIO module for multi-finger gestures. It maps these to different configurations of mouse events, depending on the number of fingers.

Parallel Operations

The first two stages of multi-touch integration already enable the use of tools via multi-touch gestures with one hand at a time. For two-handed control it was necessary to get rid of exclusive event ownership, a legacy of the single pointer, and open up event handling to parallel input streams. In Blender, input events are passed to *operators*, which are the entities that execute changes in

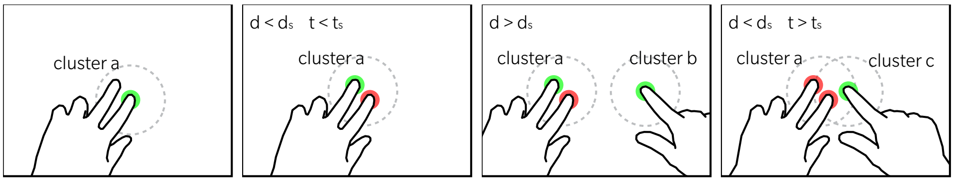


Figure 5.4: The bimanual variant of the *Chording* mouse emulation technique uses temporal (t_s) and spatial (d_s) thresholds to aggregate multiple fingers to single input streams.

the scene. Continuous input, such as from a drag-and-drop mouse gesture, is handled by *modal* operators, which swallow all incoming events and check whether they are applicable to the operation until they are aborted or complete execution. While this modal event handling is sufficient for sequential operation via a single pointer, in order for two input sources (two mice or two hands) to operate independently, multiple operators must be able to work in parallel.

This was solved by matching the IDs of incoming events with the IDs of operators. At invocation, an operator is tagged with the ID of the event calling it. Subsequent events with this ID are then exclusively passed to this operator:

```
for (handler = handlers->first; handler; handler = nexthandler)
    if (handler->op->eventid == event->id)
        ...
```

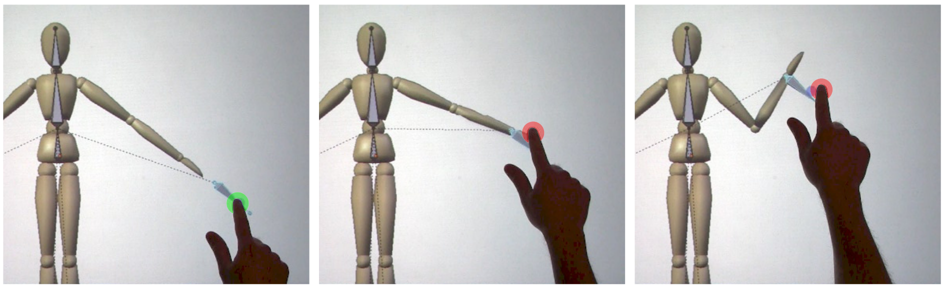
Thus, events originating from one input source are paired to a specific operator and cannot influence prior or subsequently created operators, as they are in turn paired with different input event sources via ID. Events receive their IDs in the TUIO module, ensuring that every ID is associated with one (continuous) multi-finger gesture.

5.3.3 Basic Controls

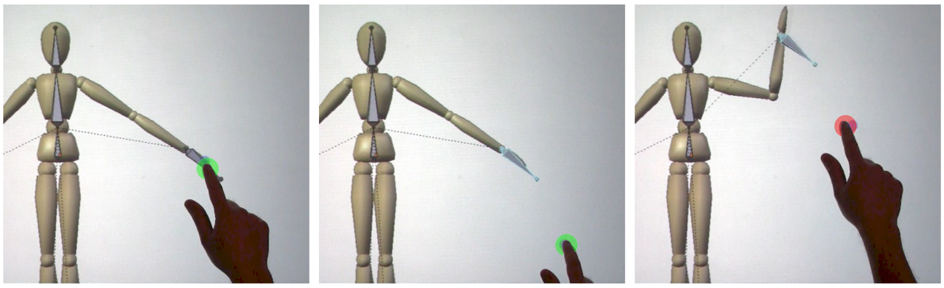
Unimanual chorded multi-finger gestures were designed for performance, camera and time control. In Blender's *3D View* window, one-finger gestures map to performance operators, while two- to four-finger gestures map to camera operators. Blender's *Timeline* window features playback buttons and a timeline, which provides temporal access control with one and timeline configuration with two to four fingers. Except for the selection (3D View) and playhead (Timeline) operators, these controls are position independent, they can be performed anywhere within the respective window.

Performance Control Interface

Performance controls use Blender's standard selection and translation operators. The translation operator works along the two axes defined by the view plane.



(a) Direct control in a single drag motion starting on the target using the tweak gesture.



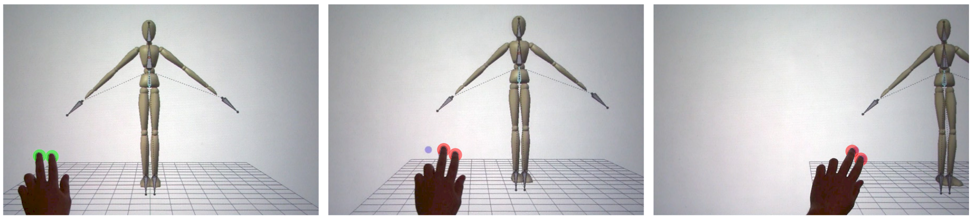
(b) Indirect control requires selecting a target prior to dragging anywhere on the screen.

Figure 5.5: Direct and indirect performance control.

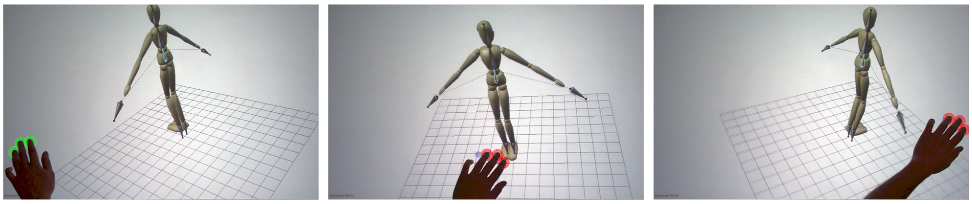
Single finger clusters produce events that map to selection (tap) and translation (drag). In linked feature hierarchies such as skeleton rigs, the translation is applied to the distal bone end, rotating the bone around screen z.

Direct control with coupled selection and translation is enabled via the *tweak* gesture. This is invoked for any drag motion that is not already assigned to an operator. With the tweak control assigned to translation, this enables selection-and-translation in a single fluid motion (figure 5.5(a)). Alternatively, the drag gesture can be performed anywhere on screen, also allowing indirect control. The currently selected feature is then translated by the absolute amount that the touch moves along screen x and y. Indirect dragging thus requires prior selection to determine the input target (figure 5.5(b)). Selection is the only context-dependent operator, as it determines the target by ray casting from the tapped screen coordinates.

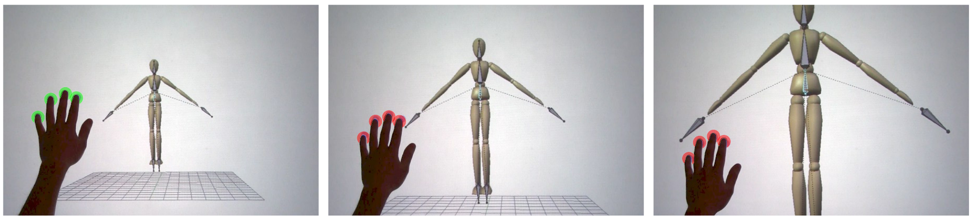
Layered animation is supported via absolute and additive blending. Absolute mode is the standard, additive mode must be activated via the GUI. Absolute control overwrites any previous transformation at the current time. In the absence of parent motion this ensures 1:1 correspondence between input and output. With parent motion, control becomes relative to the parent frame of reference (local). Additive layering preserves existing motion and adds the current relative transformation to it. By changing the view between takes so that the input-output mapping affects degrees of freedom that could not be af-



(a) Two fingers pan the view.



(b) Three fingers rotate the view.



(c) Four fingers zoom the view.

Figure 5.6: Basic view transformations with continuous multi-finger gestures.

fected in previous takes (e.g. by orbiting the view 90 degrees around screen y), this enables the animator to add depth and thus create more three-dimensional motion.

Camera Control Interface

The three camera operators *pan*, *orbit* and *zoom* map to two-, three-, and four-finger gestures. Assigning chorded multi-finger gestures to view operators does not have any precedent in the real world or prior work, and there are good arguments for different choices. A sensible measure is the frequency of use of a certain view control, and thus one could argue that more commonly used functions should be mapped to gestures with less footprint, i.e. fewer fingers. Camera dolly move or zoom is probably the least used view control, which is why it was mapped to the four finger gesture: users can zoom in and out by moving four fingers up or down screen y. Three fingers allow camera orbit by the turntable metaphor: movement along the screen x axis controls turntable azimuth, while motion along screen y controls camera altitude. Two fingers

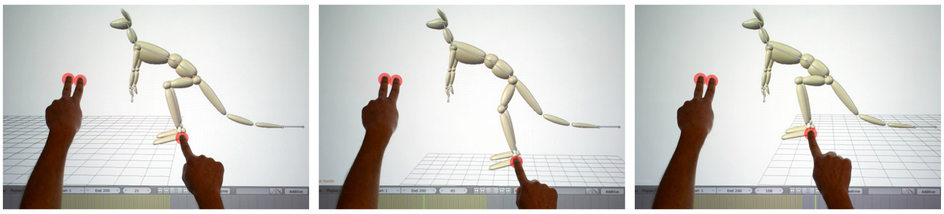


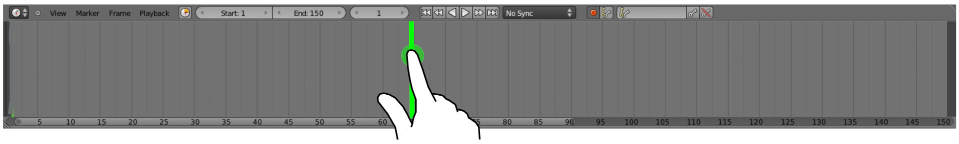
Figure 5.7: The view attaching technique. Features can inherit motion from parents animated in previous motion layers. In such cases direct control is not possible. By attaching the view to the feature's frame of reference, direct control is reintroduced.

pan the view along view plane x and y axes. Camera controls are context-free, meaning they can be activated anywhere on camera view (figure 5.6).

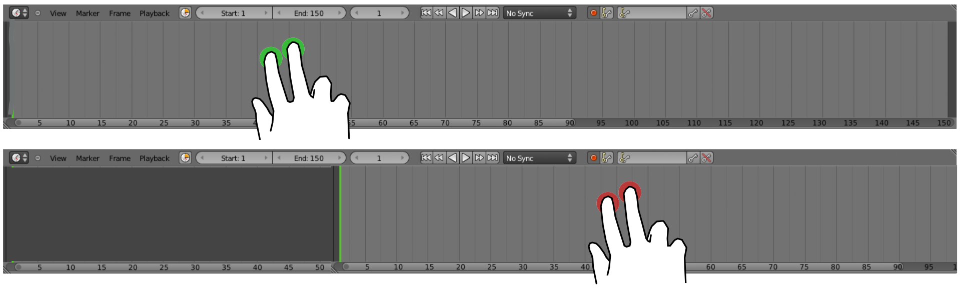
A view attachment mode, when active, fixes the view camera to the currently selected feature during all camera operations, moving the camera along with dynamic targets. The camera-feature offset is maintained and can be continuously altered depending on camera operator as described above. After establishing the attachment by starting a view control gesture, new targets can be selected and manipulated. Releasing the camera control immediately ends the attachment, rendering the camera static. By combining one-handed view control and capture in an asymmetric manner, this approach can solve indirection in control of dynamic targets (figure 5.7).

Time Control Interface

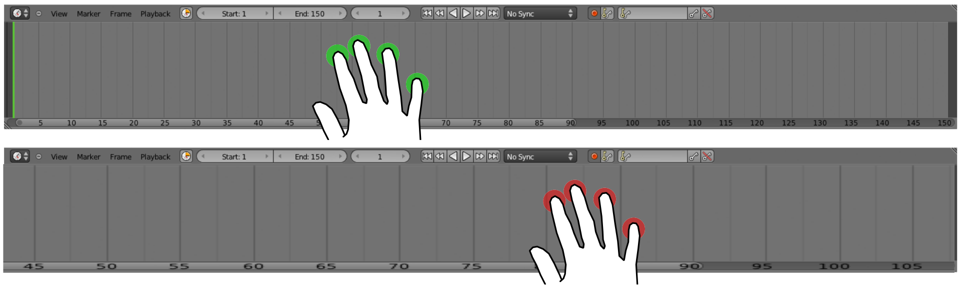
The time control interface features several buttons and a timeline (figure 5.8). Play/pause toggle buttons start and stop playback within a specified time range. A timeline gives the animator visual feedback on the remaining loop length in multi-track capture, supporting anticipation. It also enables efficient temporal navigation: with a one-finger tap the animator can set the playhead to a specific frame. A continuous horizontal scrubbing gesture allows for interactive playback speed control. Interactive playback can even be used during capture instead of a constant clock-based playback, as demonstrated by a participant of the study (section 5.4). The time range displayed in the timeline can be changed with multi-finger gestures. These temporal operations are conceptually similar to the spatial operations in view control (gesture *reuse*, cf. Wu et al., 2006): horizontal movement of two or three fingers moves the currently displayed time window back or forward in time, an up/down motion of four fingers expands/contracts the time window (cf. view pan and dolly move, respectively).



(a) One finger sets the playhead to the current frame



(b) Two or three fingers shift the displayed time range.



(c) Four fingers expand or contract the displayed time range.

Figure 5.8: Time control interface.

5.4 Evaluation

For a first impression of how people would use the system, an informal user study was conducted. Aspects of interest were the reception and use of single- and multi-track capture and camera controls, specifically in how far two-handed interaction strategies would be employed in these.

5.4.1 Design

Since the direct animation system has a high novelty and is still at prototype stage, a formative evaluation was chosen in order to guide further research. Formative evaluations are common in research and development of 3D user interfaces (Bowman et al., 2004). As this was the first evaluation of the system an informal approach was chosen: critical incidents, user comments and reactions would give valuable insights at this stage.

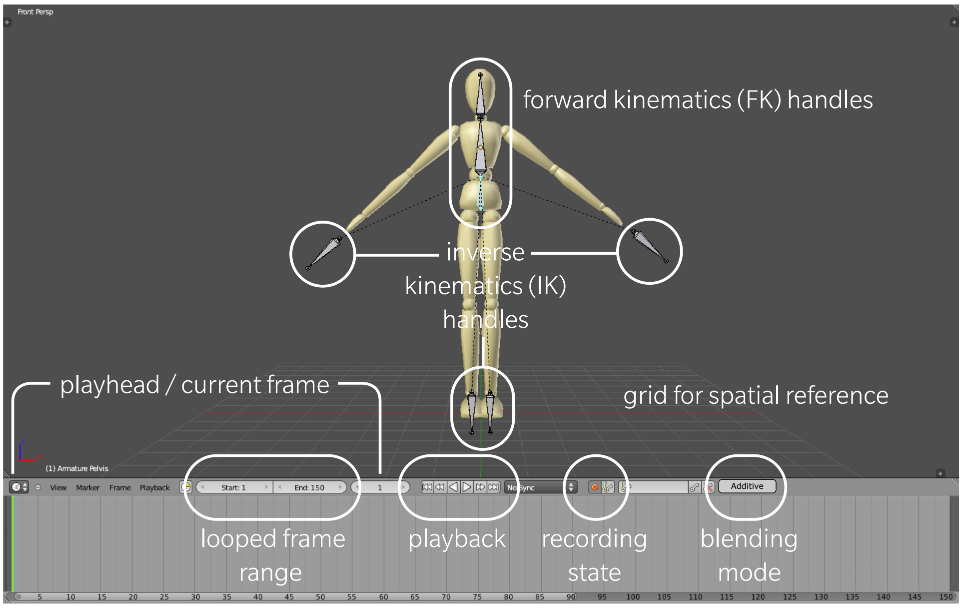


Figure 5.9: The GUI used for the free animation task was created by customising Blender's interface.

Task

The task was free animation of a stylised human puppet. An articulated mannequin was rigged with seven handles that provided puppetry controls: Three kinematic bones for control of the body—a root pelvis bone and linked torso and head bones—and four inverse kinematic handlers for hand and foot end effectors (figure 5.9). The inverse kinematics handlers allowed expressive control of the multi-joint limbs while keeping complexity at a minimum. The goal was to explore what own animation goals users would come up with given the digital puppet, so there was no need in an explicit animation task.

Apparatus and Interface

The system was tested on a rear-projected horizontal interactive surface employing the diffuse illumination technique (Schöning et al., 2008). The tabletop has a height of 90 cm and a screen diagonal of 52 inch. The rear-projected image has a resolution of 1280 × 800 pixels. Two cameras with 640 × 480 pixels resolution each give an approximate virtual camera resolution of 640 × 900. Proprietary tracking software detects finger contact blobs in the camera images and communicates these via TUIO to Blender. The Blender user interface was configured to show a 3D view and timeline window (figure 5.9).

Participants

Six right-handed individuals aged between 23 and 31 years, four male, two female, took part in the study. All came from a computer science and/or media production background. Two of these judged their skill level as frequent users of animation software, one as an occasional user and three as rarely using such software. Participants with varying skill levels were encouraged to participate in order to find out whether and how novice and expert users differed in their experience. None of the participants had yet used a performance animation system.

Procedure

Each session took about 30 minutes. The first half was dedicated to general object and view operation, with a free modelling task. The second half introduced time controls and the concepts of multi-track capture, with the free animation task described above. Of the first half, only the general aspects of selection and manipulation relevant to the animation system are reported. The specifics of the study on the modelling techniques were not relevant to the animation functionality and are documented elsewhere (Herrlich, 2013). For each half, participants were introduced to the functionality and guided in their exploration with a simple bouncing ball scene (5 minutes), before receiving the actual task (10 minutes). This included the instructor explaining verbally how to control view, space and time and asking the participant to explore each function until they had a good idea of how it worked. The instructor was careful to give only a verbal introduction and a general explanation of the available functionality in order to avoid bias. Instructions were given on how to operate individual controls, but not on which hand to use for what, or whether and how to use both hands in combination. During the free modelling/animation task the instructor did not intervene, only answering questions or adding verbal clarifications when problems arose.

5.4.2 Results

An instructor observed each participant, taking notes on user behaviour and comments as well as critical incidents.

Observations

Considering the short time frame, participants took to the controls easily. Most stated that they enjoyed using the system.

The performance control interface was straightforward for initial animations. However, dynamic targets were not easily dealt with, as participants tried to “catch” them with their fingers. Over time, some were able to address this by a combination of pausing the animation for selection and using indirect control. Multi-track animation was mainly used to animate separate features in multiple passes, less to adjust existing animation. The more complex additive mapping was hardly used and met with initial confusion, although explanation and experimenting usually solved this. One in two participants commented on being overwhelmed by the 3D manipulation task, even though they only controlled 2 DOF. Also, further editing controls were demanded, such as undo or delete for erasing previous capture takes.

The view controls were quickly understood and were used without difficulty. A problem occurred in one session when the participant did not spread fingers enough for the tracker to distinguish a pair of close fingers from a single input. After this explanation the view controls were used correctly. The most commonly used camera operation was orbit.

As all participants were familiar with the timeline metaphor they had no problems understanding it. Unfortunately, the play button was too small to allow easy operation. Combined with a slight tracking offset this caused some irritation as playback control was sometimes impaired. Most subjects easily employed the absolute positioning of the playhead to jump to a frame and to scrub along the timeline to review the animation they had created. One participant used the timeline for a method of animation somewhere between performance and frame-based animation: using the left hand for playhead and the right for pose control, he exerted a fast, efficient pose-to-pose animation style. As he proceeded, alternating between setting the playhead with one hand and adjusting the position in the other became so quick that it was more like a capture take with continuous manual playback speed control.

Five out of six participants manifested asymmetric bimanual styles of interaction. An emergent strategy of half of the participants was to assign the left hand to view or time controls and the right to capture. One participant controlled two puppet features simultaneously. Three used their left hand for a pan gesture to attach the view for animating mannequin limbs once they had already created animation for the pelvis. The benefit of locking the view to a frame of reference in this way seemed immediately apparent to them, and was greeted with enthusiasm in two cases. The near-concurrent control of playhead with the left and character feature with the right was discovered by just the one participant.

Task Outcome

Given the short timeframe and lack of experience in performance animation, participants were able to create surprisingly refined character motion. Four were able to create expressive character animations within the short timeframe of 10 minute in the free animation task. These were a walk, jump and squat motions and a "stayin' alive" disco dance move (figure 5.10). The outcomes of the two remaining participants, while not entirely random, did not display a recognisable motion indicating that they were not able to realise their vision.

Figure 5.10: Stills from animations created by participants of the study.

5.4.3 Analysis

Inexperienced users had a harder time in comprehending spatial relationships, while those with more experience in 3D animation notably picked up controls more fluently. This comes as no surprise, as using and controlling software takes time and practice, regardless of interface. For novice and casual users, the 2DOF strategy seems appropriate, since it constrains manipulation by the depth dimension. However, the interface might need improvement visualising these constraints and giving more hints on depth cues

The study indicated that the two variants of animation layering have their distinct challenges. The concept of multi-track capture of separate features was easily grasped, however the anticipated problems with dynamic targets in feature hierarchies were confirmed, and the positive reactions to the view attaching feature underline the expected benefit of this technique as one solution to this problem. Layered capture for refinement of prior capture, such as enabled with additive mapping, requires a better incentive for use in specific tasks. The free task did not offer this incentive, so no statements can be made on this aspect of multi-track animation.

The study shows that even non-experts can quickly learn to employ bimanual skills in surface-based (tabletop) 3D interaction. A learning effect was observable in that users increasingly employed both hands toward the end of the session, while one participant only used his right hand. This reflects the difficulties involved in transferring bimanual skills from the real world to the digital world, as observed in many other studies (sections 4.3.2, 5.1.6). All three of Guiard's principles of asymmetric bimanual behaviour were observed:

participants employed the left hand for view control, a spatially and temporally lower-frequency and higher amplitude task, which took precedence to and defined the reference frame for right-handed performance control. There was also some indication that the same can be applied to time operation; this is an interesting aspect for future work.

5.5 Discussion

The construction of a general-purpose animation system for direct-touch devices makes several contributions to design and engineering knowledge (section 5.5.1). Limitations of surface-based control of 3D content are discussed in section 5.5.2, before the next steps on the research agenda are outlined in section 5.5.3.

5.5.1 Contributions

The main contribution of this chapter is a design approach for a direct animation system that was validated by a proof-of-concept prototype. This included robust, easy to understand, and conflict free unimanual mappings for performance and view control. The bimanual view attaching technique enabled direct control even for dynamic targets. A user study verified the design approach by showing largely positive user reactions. The majority of users employed both hands in emergent asymmetric and symmetric bimanual interaction.

The development approach gave insight on adapting WIMP input architectures to concurrent multi-point input. The software used as the basis for the prototype was designed for single-focus, single-pointer interaction rather than parallel interaction. Three modifications of the WIMP event system addressed this. An input interface using a standard protocol layer internally exposed finger contact data. Aggregating finger contacts with the chording technique resulted in distinct 2DOF input clusters that are compatible to legacy tools. Breaking strict modality for operators was necessary for ensuring parallel bimanual operations.

5.5.2 Known Limitations

The design approach dealt with several tradeoffs: low- versus high-DOF multi-touch controls, direct-touch versus tangible interfaces and planar versus 3D input.

2DOF mappings were chosen since they follow the structure of planar input devices and because they are compatible to multi-finger mouse emulation. Sections 2.1 and 5.1 discuss recently proposed techniques that use further degrees of freedom of hands and fingers in multi-touch input, such as distance between fingers, in order to better match target DOF. Yet so far these have only been applied and evaluated for manipulation tasks in which the result counts, as opposed to performance capture, where the process of the manipulation counts.

It remains to be seen how well these higher-DOF techniques are suited for tasks requiring continuous coordination of many DOF.

Tangible user interfaces that integrate physical and digital representation (Ishii and Ullmer, 1997; Ullmer and Ishii, 2000) are possibly the most direct class of interface. Tuddenham et al. (2010) showed that physical props augmenting intangible projected representations provide significant advantages over multi-touch input in acquisition and manipulation tasks. For instance, they can improve control strategies and eliminate exit error, benefiting task performance. However, it is yet unclear how such physical augmentations of digital displays can work for general purpose animation control. Digital mappings can constrain DOF based on changing models, while physical props have a single, built-in model (Esposito et al., 1995; Knep et al., 1995; Jurgensen, 2008). For layered animation such props would require precise actuators that would need to be able to reproduce the input motion applied by the performer.

While interactive surfaces are not truly tangible interfaces, they do offer a basic haptic constraint as opposed to free-space devices. These support 3 to 6-DOF tracking or more, enabling integrated control closest to what we know from real three-dimensional space, but suffer limitations that stem from the lack of tactile feedback. Grossman and Wigdor (2007) point out that free-space input lacks discrete input via contact-to-surface events. Discrete input via clicking, pressing, tapping is established and widely used. Without physical forms to provide resistance, free-space interaction needs to support recognising discrete gestures such as finger snapping in order to provide for discrete trigger signals. Hinckley et al. (1994) find that physical constraints are superior to software constraints, since they provide a form of feedback that can be (haptically) experienced rather than just optically, requiring less mental effort. Surface-based input can provide such hardware constraints to a certain degree if interaction designers build interfaces that exploit the surface nature of input. Experiments by Bérard et al. (2009) found surface-constrained input mechanisms superior in precision and task completion in an indirect object placement task. They explain this by hand instability and increased user fatigue when users cannot support their hands and body, raising the further problem of ergonomics with free-space input. The correct choice of input device must thus always find a tradeoff between degrees of freedom and the benefit of physical constraints and their positive effects on precision and ergonomics. The studies cited above suggest that the benefits of surface constraints can outweigh the benefits of higher dimensionality of input.

5.5.3 Research Agenda

In future work, performance controls need to be extended by rigid body rotation and scaling to accommodate more motion design scenarios. Orientation and size could be controlled by handles, breaking the direct manipulation metaphor, or by borrowing from high-DOF techniques such as *Sticky Tools* (Hancock et al., 2009). While the approach presented chose kinematic mappings, integrating simulation of local (Oore et al., 2002b) or global (Laszlo et al., 2000; Zhao and van de Panne, 2005) physical forces to aid the creation of object or character movement is certainly worth investigating. Specifically, the interplay of direct-touch physical mappings (Cao et al., 2008; Wilson et al., 2008) and the animation requirements of precise control and expressiveness pose challenges for further research. Feedback improvements include visualising mapping constraints, or adding depth cues for improved perception of projected 3D spaces. Finally, the assumed benefits of low- over high-DOF surface transformation controls still needs to be confirmed in comparative studies.

Explicit visualisations of multiple recorded tracks could improve multi-track control. These could either be in-view, rendering ghosted versions of previous takes into the scene for better anticipation and synchronisation, or in an external sequencer interface (cf. Svensson et al., 2008). In-view channel visualisation could also be the foundation for alternative layering mappings next to absolute and additive, such as channel averaging. Further, a more in-depth investigation of bimanual and unimanual view attachment techniques could add credit to the preliminary findings presented here.

Bimanual interaction can be further exploited. Motion capture mappings could explicitly support asymmetric or symmetric control. For example, the left hand could provide constraints for the action space of the right (cf. alignment techniques in Herrlich, 2013). If multi-finger gestures are used for performance control, new mode switches must be found for view control. Left/right hand distinction could add a further means of discerning modes of operation (Walther-Franks et al., 2011a). Lastly, the study opened up the intriguing interplay of left and right hand in sequential and simultaneous space-time control. Techniques that dedicate one hand to time control and the other to capture could be interesting for frame-based and performance animation alike, and even blur the boundary between the two.

Chapter 6

An Interaction Technique for Direct Motion Timing

Timing is at the heart of animation. It gives meaning to movement, it can convey weight, size and even emotion (Lasseter, 1987). In traditional animation, timing is also the most difficult aspect (cf. Terra and Metoyer, 2004). The abstraction involved in the spatial definition of temporal events must be learnt over many years. In performance animation dynamics can be described more intuitively by acting out motion in real time. Performance timing techniques only apply this to the timing of an existing animation, essentially separating the temporal from the spatial description of motion. Separate timing is helpful to adapt behaviour and style, or to focus first on spacing, then on timing. Yet interaction metaphors of state-of-the-art performance timing techniques fall short of the direct animation principles of high input-output correspondence and spatiotemporal directness.

Research on video navigation interfaces has brought forward a time control technique that features direct interaction and spatial correspondence between input and target motion: direct manipulation video browsing enables temporal navigation by dragging a moving feature "through time" along its trajectory (section 2.2). This chapter presents *Dragimation*, a new technique for direct motion timing that uses the concept of time control through a direct spatial mapping. A study comparing Dragimation to state of the art performance timing techniques illuminates how it improves precision and user satisfaction in timing tasks. Both objective measurements and subjective user rankings are highly in favour of Dragimation for performance timing[1].

After a treatment of the design space of time control interfaces in section 6.1, a new approach to performance-based motion timing called Dragimation is developed in section 6.2. Dragimation is compared to other performance timing techniques in section 6.3. A summary of contributions, discussion of limitations and outlook on future work in section 6.4 conclude the chapter.

---

[1] Parts of this chapter have been previously published (Walther-Franks et al., 2012)

6.1 Design Space of Motion Timing Interfaces

Both in traditional animation and performance animation, timing is normally defined as part of the motion. However, timing can also be treated separately to spatial aspects of motion specification. Keyframe animators might first create key poses and then temporally arrange them to suit the desired dynamics. Motion capture data might be temporally adjusted to synchronise the animations of a character relative to the timing of another. The prevalent approach to temporal editing is to use time plots (cf. section 2.2).

In performance animation, blending several capture takes allows a certain spatial adjustment (cf. section 5.1.4), but is not suited for retiming. Performance timing applies captured input motion only to the timing of an animation; it requires the full motion to have been defined previously, either with frame-based or performance animation methods. So far two techniques have been suggested, of which only one has been scientifically published. They are presented in section 2.3.3 and summarised here for convenience. With the timeline technique, the animator interactively records a timing by scrubbing along the timeline (Sampath, 1999). The propagation through time created by input movement is recorded in real time and applied as the new timing. A sketch-based approach lets the animator describe a feature's timing by mimicking its motion (Terra and Metoyer, 2004, 2007). The sketched path is then matched to the original trajectory. The animation timing is changed so that it matches the speed at which the pen moved while sketching—slow sketching creates a slow timing, faster pen movements a faster timing. After a successful match the animator gets feedback on the result.

A further class of time control techniques was recently established in the domain of video browsing. Instead of abstract spatial metaphors, direct manipulation video navigation uses the trajectories of video features as a time control handle. Such techniques neglect the time domain when interacting with a scene that is already timed. They typically show the motion path, and provide a way to navigate along this path by dragging the object to the desired point. Direct manipulation video navigation (DMVN) is discussed in section 2.2 and summarised here for convenience. *Trailblazing* employs direct manipulation in a video surveillance setting to interact with objects in the video or on a floor plan (Kimber et al., 2007). *DRAGON* (Karrer et al., 2008) and *DimP* (Dragicevic et al., 2008) propose a more general use of direct manipulation browsing for video analysis, e.g. of sport videos. Goldman et al. (2008) showed the feasibility of this approach for a variety of video editing tasks. Shah and Narayanan (2011) also used direct manipulation for video editing, retiming a segmented part of the video against a still background or the rest of the scene.

In the rest of this section, frame-based and performance-based timing techniques and DMVN are analysed using the design space of chapter 3 and the direct animation guidelines of chapter 4.

6.1.1 Task

Time control tasks can be divided into non-invasive navigation, in which the user explores time in order to search for a specific event or browses in order to receive an overview of events, and invasive timing operations, which change the temporal arrangement of events. Techniques based on time plots allow time-based navigation and timing, meaning that the browsing or editing instruments do not offer any spatial information. Techniques based on sketching trajectories can be used for spatial navigation and timing, although they lack real time feedback. DMVN offers content-based time control, which directly connects the time control instrument to the spatial extent of motions. Proponents argue that browsing or search tasks are often defined by spatial parameters, for which this technique is far better suited (cf. Karrer, 2013). In motion design tasks, spatial and temporal parameters are also highly integrated. These direct manipulation techniques have thus far been hardly considered for temporal editing. While the performance controls of the direct animation system offered integrated spatiotemporal control, performance timing supports the principle of *separate temporal control*. This gives more possibilities for motion editing, which is as important as motion creation (guideline 1).

6.1.2 Metaphor

Time control interfaces commonly employ spatial metaphors. These range from abstract timeline metaphors to more literal path- or object-based ones. Time plots rely on culturally dependent abstractions rather than drawing upon users' innate environment awareness and skills or body awareness and skills (violating guideline 8). Path descriptions as a global motion description abstract from local timing. Object-based time control uses the direct manipulation metaphor, conforming to guideline 9. Since DMVN requires precise, predictable dragging behaviour, kinematic mappings as used for regular drag controls are the first and obvious choice (although physics-based controls are conceivable, see 6.4.3).

6.1.3 Directness

Analysing directness, one can discuss the spatial offset from input to targets and the temporal offset between input actions and the moment in which their effect is visible. Time-plots used in frame-based animation and the performance-based scrubbing necessarily introduce a spatial offset, since the area of input is the spatial metaphor. Temporal indirection is low since manipulations (a moved keyframe, the scrubbed playhead) immediately update the scene view[2]. The sketch-based performance timing approach works best if the input path is sketched in proximity to the motion path. Since the result is not visible until after a suc-

---

[2] This requires real time scene updating, which in complex scenes is not always given despite the rendering power of current workstations.

cessful path match, the temporal offset is large. DMVN offers the lowest spatial and temporal offset, since the dragging operation occurs on the current view of the scene, which is immediately updated on user input. Of the above techniques direct manipulation time control thus best fulfils direct animation guideline 3.

6.1.4 Correspondence

Correspondence between input and output (guideline 7) is also a desirable trait for time control interfaces. The abstract control-to-content mapping underlying time plots offers no correspondence between input and resulting scene motion. High correspondence is beneficial for the sketch-based approach to performance-based motion timing, since sketches that closely resemble the motion path are most easily matched. The dragging mechanism of DMVN requires the input motion (cursor or the likes) to be as close as possible to the motion path, as deviating from the path can cause jumpy navigation (cf. Dragicevic et al., 2008).

6.1.5 Space-Time

Presentation dimensions can be edited integrally during generation or spatiotemporal editing, or separately with spatial and temporal editing. Integral creation and editing techniques map user input to medium space-time, locating them in the second column of the matrix classification scheme (*input*→space-time), while spatial editing techniques are in the first (*input*→space) and timing techniques in the third column (*input*→time).

Likewise, either the spatial, temporal or both components of input can be applied in the mapping. Frame-based techniques rely on static controls that neglect the dynamics of user actions (space→*output*). An example of only spatial control is manipulating individual key poses or the whole motion graph (space→space). Frame-based timing interfaces use graphical representations, such as the timeline or the dope sheet (space→time). If these plot spatial parameters to time they can also be used for partially integrated spatiotemporal editing (space→space-time).

Performance animation techniques draw on the dynamics of input motions to control virtual time (time→*output*). Performance controls define spatiotemporal relations integrally; layered acting also allows editing motion in this manner (space-time→space-time). Performance-based timing techniques read the dynamics of spatial input to redefine medium timing, enabling separate temporal control (space-time→time).

6.1.6 Integrality and Manuality

Since time is one-dimensional, time control requires only one input DOF. For two-dimensional time plots, the second control dimension is often used to manipulate one spatial parameter in a function graph. Scrubbing controls can benefit from a scrubbing speed/timeline scale control on the (second) orthogonal

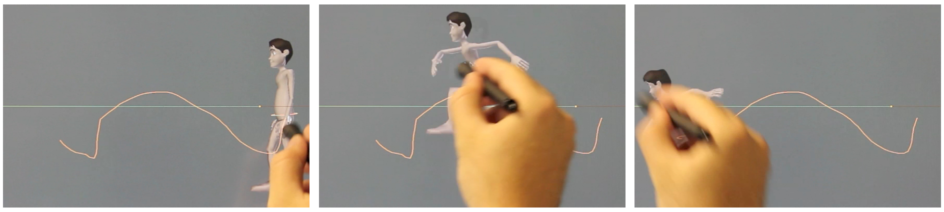


Figure 6.1: Timing a leap animation with Dragimation.

spatial input dimension (Hürst et al., 2004). Since translation input suffices for dragging operations, sketch-based timing technique and DMVN only require 2-3 input DOF, depending on whether content is 2D or 3D. With appropriately relaxed DOF requirements, the techniques at hand can fulfil guideline 5.

Guideline 10 does not apply to the timing techniques, as they require only one hand to operate. Bimanual operation can however make sense in combination with other functionality, e.g. view navigation (chapter 5).

6.2 Direct Motion Timing with Dragimation

Dragimation brings direct manipulation to performance-based animation timing. It combines the strengths of timeline- and sketch-based performance timing techniques into one, using direct manipulation and interactive feedback to give the animator a sense of space and time. The local, real time input-output mapping eschews any extra post-hoc matching and time control based on spatial phenomena ensures high correspondence between input and result.

With Dragimation, the animator drags the object to be animated along its motion path through the scene. This is a direct manipulation interaction; feedback is immediately given, allowing closed-loop performance adjustments. On releasing the drag operation, the animation is already fully re-timed. This approach has two advantages: Firstly, it reduces the global matching problem between two representations of a path—the one carefully laid out in the animation and the one drawn by the user for re-timing—to a local problem in which at any point during the process an input position simply has to be projected onto the animation path of the object. This local problem has been studied in the area of DMVNs and can be solved in a variety of ways. Such mappings require a close coupling of input motion to object motion, strengthening the directness of the interaction. Secondly, the resulting animation is already visible even during the interactive re-timing by the motion of the object itself. This removes a layer of cognitive and physical indirection that especially novices have difficulties with. In contrast to performance timing via timeline scrubbing, Dragimation maintains a correspondence between input and output motion.

6.2.1 Input/Output Mapping

In order to retime an animation, the animator picks a feature and drags it along its motion path through time. Visual feedback is updated according to the current input, giving the animator an immediate feedback on his actions. When releasing the drag, the animation is retimed to represent the timing that the animator has just acted out by dragging.

It is assumed that the user has set a view so that the trajectory is maximally parallel to the view plane, i.e. with no segments of the path at a large angle to the view plane, for a near-constant 2D input to 3D motion ratio. It is further assumed that the correct sequence of key poses is already set, so that the feature only moves forward along the trajectory. For the algorithm described below it is easier to work with a set of sampled curve points than with an analytical description of the curve. Samples need to be drawn at regular spatial rather than temporal intervals in order to stay independent of the original timing.

The simplest metric for determining the current position along the curve is to find a point on the curve with the shortest distance to the input cursor. However, with complex paths or fast movements this can result in unwanted jumps along the curve. For timing it is critical that an animator can create fluid movements along the curve. Thus the *arc length continuity* formulation of Dragicevic et al. (2008) is used, which extends the distance metric to include the change in arc length as a third dimension.

$$D = \sqrt{(p_x - C_x)^2 + (p_y - C_y)^2 + (k \cdot \overline{C_a C})^2} \tag{6.1}$$

p_x and p_y are the coordinates of the pointer on the screen, C_x and C_y are the coordinates of any point C on the curve projection, and $\overline{C_a C}$ is the arc-length distance between the currently active point C_a and C on the projected curve (figure 6.2). The scalar $k \geq 0$ allows to weight the arc-length continuity component. While Dragicevic et al. recommend a value of $k \approx 1$ for good results in video navigation, this produces too much "slur" or lag for the performance timing task, especially with fast animations. $k \approx 0.5$ yields a good tradeoff between an interactive response and smoothly following the path. Travel along the curve is restricted to one direction by only searching the segment of the curve between the currently active point C_a and the far end of the curve. This direction constraint guarantees *directional continuity*, so that it need not be explicitly considered in the metric. In order to create a true dragging action, time traversal is started when the cursor is in proximity of the feature, rather than enabling a user to click anywhere on the curve.

6.2.2 Motion Retiming

Dragimation records a relation between real playback time and the existing source time that is stored in a list of tuples (source time, playback time) called a *time map*. Since the time map initially only covers the frame range visited

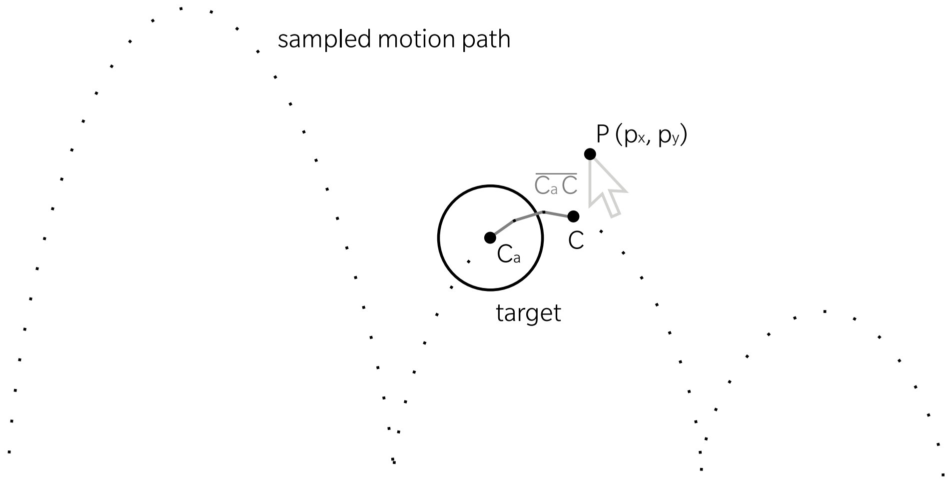


Figure 6.2: Illustration of the distance metric used in Dragimation (eq. 6.1).

during interactive timing and the recorded playback (or real) time is not yet aligned with the animation time, some post-processing on the time map makes it ready for look-up:

- The two time domains are aligned by offsetting the playback time component of each tuple by the source time of the first recorded tuple.
- Frames before the retimed range are covered by adding a tuple (t_0^{src}, t_0^{src}) to the beginning of the list, where t_0^{src} is the first frame of the source time.
- Frames after the retimed range are covered by appending a tuple $(t_1^{src}$, $t_1^{pb})$ to the end of the list, where t_1^{src} is the last frame of the source time range and t_1^{pb} is the last frame of the playback time range (the same value adjusted by the temporal contraction or dilation of the retimed range).

To retime an animation's (key) frames, the source time mapped to each frame's playback time is looked up and set as it's new time. If there is no tuple in the list with a keyframe's playback time, the value is interpolated linearly between the nearest value before and the nearest after the frame. Since movement is constrained to only forward in time, each source time is only assigned a single playback time value (the derivative of the mapping function is never negative).

Terra and Metoyer (2004, 2007) identify a problem with this method of shifting keyframes in time that can occur for certain descriptions of the underlying motion curve. Key frames are control points of a 2D spline, whose x values define time, and whose y values define a spatial parameter. Each key frame has

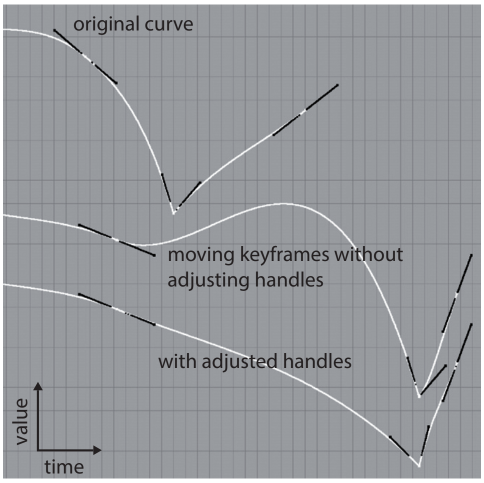


Figure 6.3: When key frames are shifted in time without adjusting spline handles, this can change curve tangency and thus the spatial configuration of the motion path. Image adapted from Terra and Metoyer (2007)

.

two manually adjustable handles for defining the tangent at the control point. If the key frame is shifted in time but the handles are fixed relative to it, this can result in undesired changes in the curve that no longer represent temporal contractions or dilations, but also cause changes to the feature trajectory (figure 6.3). The trigonometric procedure of Terra and Metoyer to correct curve tangency is used in order to maintain the spatial configuration of the motion.

6.3 Evaluation

To substantiate claims on the superior directness, correspondence and metaphor of Dragimation, a study was conducted comparing it to the two state-of-the-art performance timing techniques. Section 6.3.1 describes the reference implementations of these techniques in detail. The study design is given in section 6.3.2, its results are reported in section 6.3.3 and analysed in section 6.3.4.

6.3.1 Reference Techniques

The timeline- and sketch-based performance timing techniques had to be re-implemented, as the software implementing the scrubbing technique is outdated, and the tools developed by Terra and Metoyer were never released. Furthermore, for the study it was desirable to integrate all three into one evaluation prototype in order to maximise comparability. For easier reference the former is termed *Scrubbing* and the latter *Sketching*.

Sketching

Sketching implements the sketch-based technique for performance-based keyframe timing by Terra and Metoyer (2004, 2007). It builds on a static representation of a feature's motion, its motion path, which can be generated by sampling the feature location at regular intervals of the animation. To retime, the user mimics the motion path by sketching a similar path with a pen on a tablet input device. Motion path and sketched path are then matched (semi-)automatically and the timing of sketch path samples determines how keyframes are retimed.

The sketched path is recorded as a list of triples (x, y, t) with two spatial and one temporal dimension. The system then determines salient points on both curves by finding local minima and maxima in both spatial dimensions. Thus detected minima and maxima are filtered further by a threshold. These salient points divide both curves into segments, and are used to match the two curves: for every keyframe, the normalised arc length position along a curve segment of the original motion curve is used to calculate the recorded playback time of the corresponding point on the sketched curve. If the algorithm does not find a good match between salient points on both curves, the salient points can be edited manually (figure 6.4).

For the study this technique was improved beyond the state of the art, as partly suggested by Terra and Metoyer (2007) after their user study. To improve the manual matching process, the source curve changes colour depending on the success of the matching: if the number of salient points matches, the curve turns green, otherwise red. The salient points were also numbered to make it easier to determine where on the path they lie.

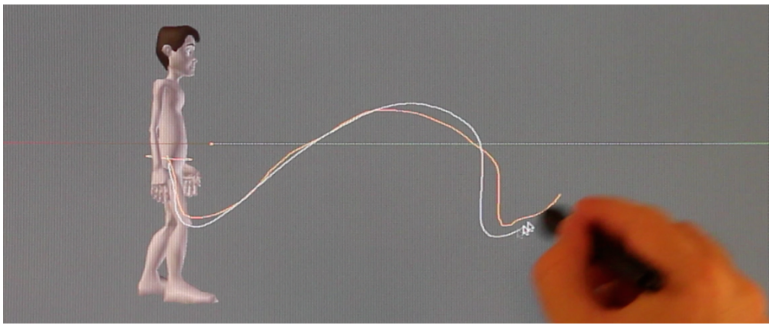
For the Sketching tool, a time map is created by matching the paths based on salient features using the procedure described in the original paper. Motion is then retimed using time map lookup as described in section 6.2.

Scrubbing

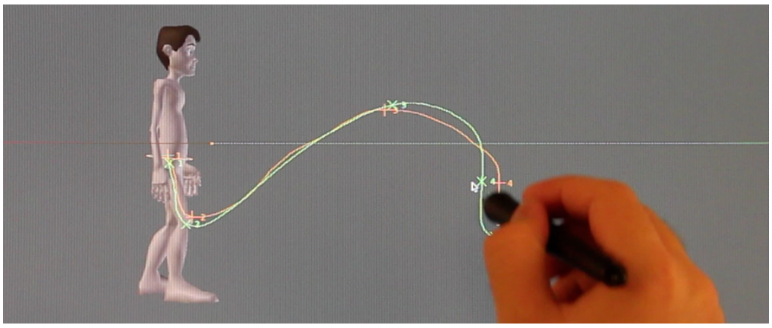
Scrubbing implements the timeline-based technique originally presented as a plugin to the 3D creative suite Maya (Sampath, 1999). Using Scrubbing, the user slides or scrubs along the timeline to define the new timing.

A linear spatial mapping relates cursor position to source time. The view is updated when a new source time is visited, giving interactive feedback. Releasing control retimes the frame range scrubbed by applying the recorded time mapping. For example, if frame range 1 to 24 was scrubbed and frame 1 was touched at 0 seconds and frame 24 at 0.5 seconds, a keyframe at frame 1 would keep its timing while a keyframe at 24 would be retimed to frame 12 $(0.5s \cdot 24fps)$.

For the study, modifications and extensions beyond the original technique were made: First, since it can be assumed that the sequence of motion events



(a) Stage one: sketch the path with a certain timing.



(b) Stage two: check result of path matching and adjust if necessary

Figure 6.4: The two stages of Sketching the timing.

over time should remain the same, movement of the playhead can be constrained to only forward in time, i.e. a movement from left to right on the timeline. Second, to be independent from frames and framerate, sub-frame scrubbing was implemented: every value of input space is continuously mapped to floating point time, rather than rounded to integer frame values. Thus, the only limiting factors are input device resolution and the dimensions of the timeline and frame range it depicts, most of which can be adjusted to meet the resolution desired.

For every frame visited during the scrub input, the playback (or real) time that has passed since scrubbing was initiated is saved together with the currently visited source time. This tuple (source time, playback time) is saved in a time map, which is then used for retiming the frame range visited analogous to Dragimation (section 6.2).

6.3.2 Design

The study design is based on that of Terra and Metoyer (2007). Since the goal was to evaluate all three performance timing techniques, a setup with three performance tools was chosen. Keyframing as a baseline was omitted since this was already treated by Terra and Metoyer.

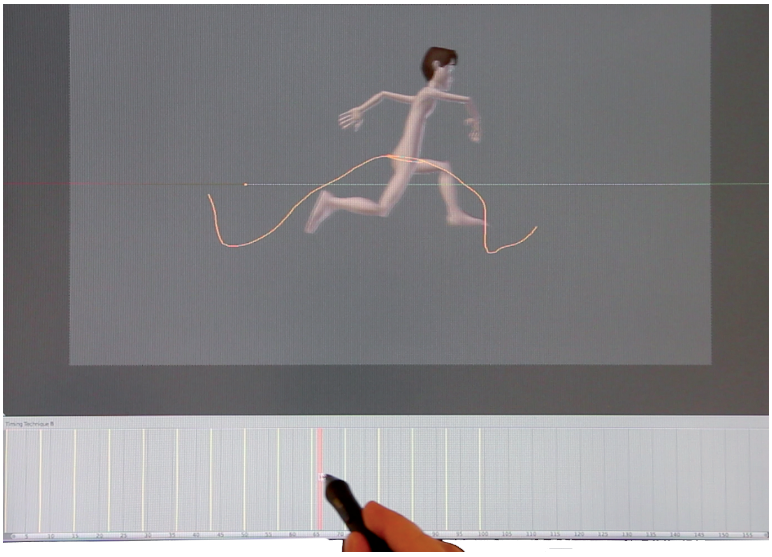


Figure 6.5: Scrubbing uses the timeline as a recording device.

Hypothesis

The hypothesis of the experiment is that the directness of the interaction and the closed-loop feedback that Dragimation provides have a positive impact on both the objective performance of the animator and their subjective preference compared to Sketching and Scrubbing.

Task

The entire workflow in which performance-based timing techniques would be used has three steps:

1. Create a motion with any desired method. The timing of this motion can be arbitrary, a rough sequence of events suffices.
2. Act out the timing. The recorded timing is applied to the temporal layout of the motion. This step can be redone until the desired timing is reached.
3. Further adjust the motion with other tools if necessary.

Since only the timing tools are under investigation, the motion layout was predefined, skipping step 1, and the post-editing process omitted, leaving out step 3. The study thus focused on step 2, the timing part of the workflow.

In order to have a comparable measure of how well a tool can be used for timing the task of timing to a reference is used. For a set of sample animations,

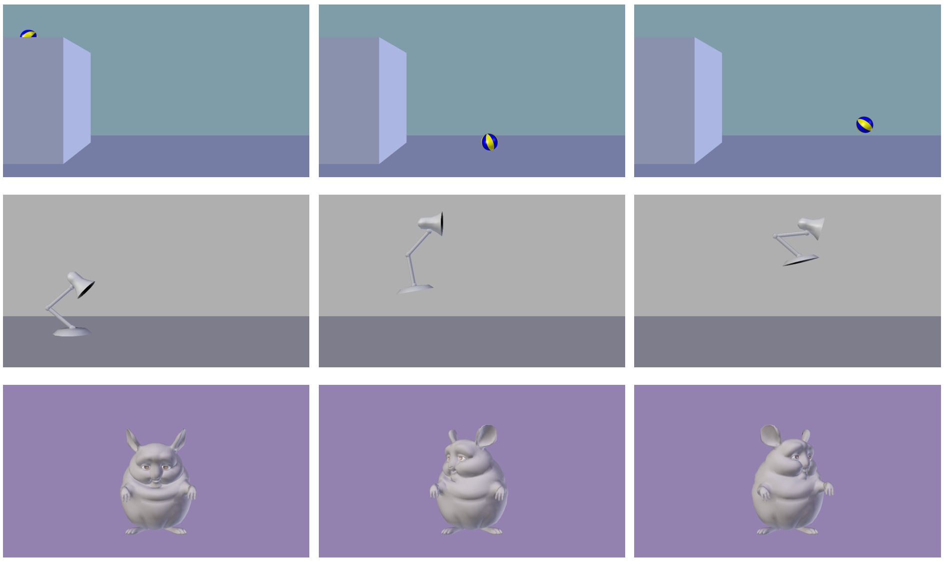


Figure 6.6: The three scenes used for the timing task in the user study. Bouncing ball, jumping lamp character, chinchilla looking from side to side.

a target timing was defined which subjects were asked to imitate as closely as possible.

Four sample animations were selected for the timing tasks, one for the tutorial and three for the main study. These were taken from the *Blender 2.5 Character Animation Cookbook* (Vasconcelos, 2011) and the *Blender 2.57 regression suite*. The criteria considered were speed of the animation, spatial complexity of the motion, and overall animation length. Also, animations were selected to be representative of typical animation tasks. The three scenes (figure 6.6) were a spatially complex bouncing ball animation of medium length and speed (task 1), a lamp character in a fairly complex short jumping animation with anticipation and follow-through (task 2), and an animal character slowly looking from side to side (task 3). The tutorial scene featured a jumping humanoid character and was of shorter length. The keyframes of each animation were changed to be distributed over time at equal spacing.

Apparatus and Interface

The study used a pen-based interactive surface, a Wacom Cintiq UX. Direct pen input was chosen over direct touch for several reasons. Pen-based devices are better established and thus better known to most users, especially digital artists, allowing the study to focus on the novelty of the technique, not the hardware. They are also technically more mature than touch-based hardware, with robust detection and a high response rate. Terra and Metoyer (2007) also recommend pen input for performance timing.

Participants were seated at a table with the interactive surface as well as a second monitor. They were asked to adjust angle and position of the Cintiq as well as the height of the chair to a comfortable position. The second monitor played a rendered video with the goal timing of the current scene in an endless loop. The user interface of the Cintiq consisted of a 3D view of the scene with the feature to be retimed and its motion path highlighted (figure 6.7). For the Scrubbing technique, a timeline was displayed beneath the 3D view, for Sketching and Dragimation this was left blank. For the Sketching technique, the bar below the 3D view featured a large button labeled "Apply". A complete trial using Sketching consisted of performing the timing once, editing the resulting curve if necessary, and manually issuing the Apply command. For Scrubbing and Dragimation, a complete trial consisted of performing the timing once, which was then automatically applied on lifting the pen from the screen. After the timing was applied, the resulting animation was played back in an endless loop for review. A tap with the pen ended the playback and reset the scene and timing to the initial setup. The current tool and trial were displayed in the upper right corner of the display.

Participants

27 subjects aged 22–43 (average=28.4) participated in the study, 7 of which were female. All were right-handed. 11 were experienced in computer animation (more than 3 years of experience), 6 considered themselves intermediate (between 0.5 and 3 years of experience) and 10 were novices to animation (less than 0.5 years of experience).

Procedure

The study used a within-subjects design, with each participant testing all three techniques on the same three scenes. First, the experimenter explained and demonstrated the use of each timing tool. Participants were then given sufficient time to explore each technique with a tutorial task until they felt comfortable using it. The main part of the study was done in three blocks, testing all three techniques on each scene. Since the tasks were quite different in nature for each scene, the learning effect between scenes was judged to be negligible. This kept the presentation order of the scenes constant, while the order of techniques was counter-balanced based on a Latin square. For each tool, participants had 10 trials to approximate the goal timing as closely as possible, resulting in 90 trials per user. A trial consisted of using the tool once and viewing the resulting timing. After 10 trials with one tool, the system automatically switched to the next tool, until all three techniques had been used for the current scene. The resulting animation and duration were recorded for each trial.

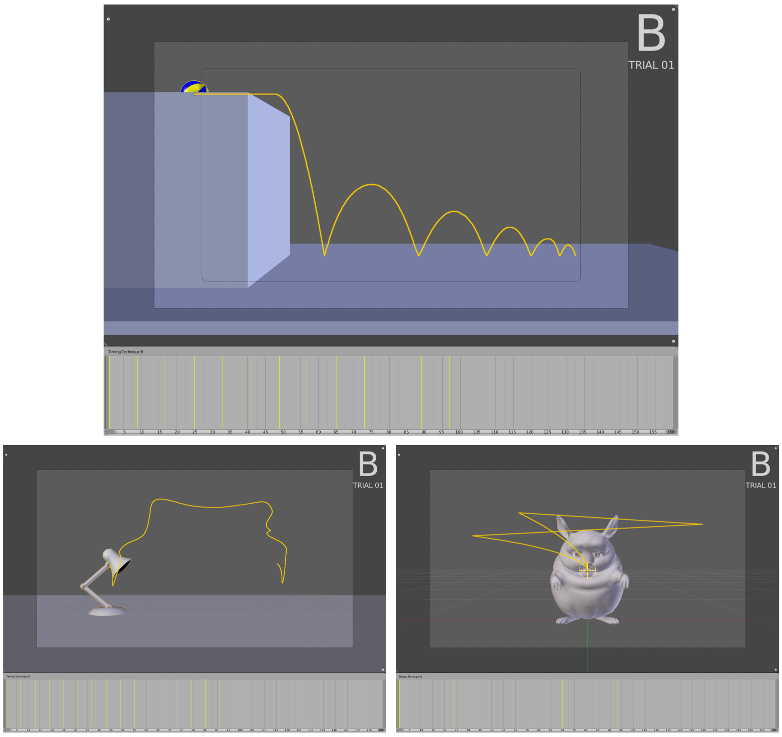


Figure 6.7: The interface as used in the study, with all three scenes. The motion path is displayed as an orange curve. The timeline was only displayed for the scrub technique, for the other two it was left blank. Current technique (coded as a letter) and trial were displayed in the top right corner.

After each scene, participants were asked to rank the three techniques according to *precision* ("With which technique did you feel you achieved the most exact result, closest to the reference timing?"), *ease of use* ("How cumbersome did it seem to you to create a timing with each technique?"), and *mental load* ("During the task, how often did you have to consciously remember how a technique works?"), plus an *overall ranking* on which tool they most preferred for this scene. They were asked to rank the techniques based only on usage with the scene they had just retimed. After all three scenes, participants were asked to comment on the learnability of each technique using the learnability sub-scale of the System Usability Scale questionnaire (Lewis and Sauro, 2009). This was followed up by an interview in which subjects were asked to remark on any positive or negative impressions, and to give a comparison to other timing techniques they were familiar with, if any. Finally, participants filled in a form on demographics and prior experience.

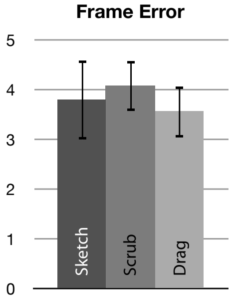


Figure 6.8: Comparison of overall frame error means. Error bars indicate 95% confidence interval.

6.3.3 Results

The results of precision and time measurements are presented, before reporting on user observation and subjective assessments.

Precision and Time

To determine quality of timing, the measure proposed by Terra and Metoyer (2007) was employed. Per trial, the offset between target and achieved time for each keyframe is calculated. Since the three animations have a different number of keyframes, the average error over all keyframes for each trial is calculated, giving a single value for proximity to the target timing. In order to account for a learning effect, the trial with the lowest error from each set of 10 trials per scene per technique was picked. After cleaning the data from outliers (video data corroborates that one subject did not follow the task of timing to a target but rather created a completely different timing, largely ignoring the reference video), the mean errors in frames are 3.85 (SD 3.51) for Sketching, 4.09 (SD 2.30) for Scrubbing and 3.58 (SD 2.03) for Dragimation (figure 6.8). A Friedman test shows a significant difference ($p = 0.007$) for technique. Pairwise Wilcoxon tests show Sketching ($p = 0.017$) and Dragimation ($p = 0.033$) to have significantly lower error than Scrubbing. A Mann-Whitney test comparing expert and novices did not reveal any significant effect.

With an average of 12.9 (SD 4.4) and 14.2 (SD 4.9) seconds per trial respectively, Scrubbing and Dragimation were approximately equally fast to use, while the timing process with Sketching took roughly twice as long with 27.4 (SD 7.6) seconds on average. This can be attributed to the fact that the sketch-based technique is not fully real time, due to its manual feature editing. In the scope of the whole animation process, this time difference is negligible, and no further distinction between the techniques can be made based on task completion time. However, it is evident that these values are far below the time required to complete the task with standard keyframe placement.

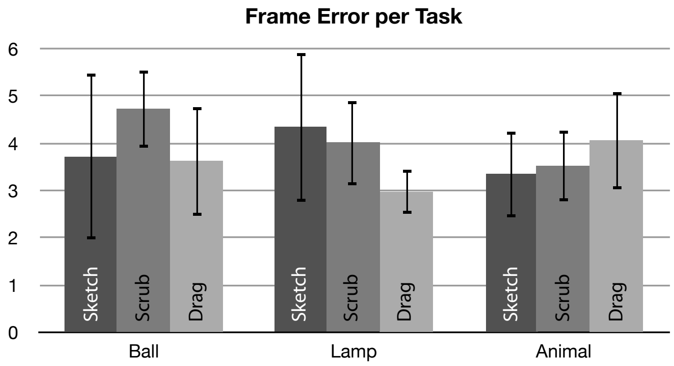


Figure 6.9: Comparison of per task frame error means. Error bars indicate 95% confidence interval.

Observation

Video footage from the experiment illuminates issues with the setup and individual techniques. The task of timing to a video presented at a nearby monitor seemed to create an artificial situation that obstructed the interactive feedback of Scrubbing and Dragimation. In many cases, subjects watched the reference video while performing the timing on the interactive display. While this worked for Sketching, neglecting the visualisation impaired using Scrubbing and Dragimation, since these rely on interactive feedback. This is an artificial condition as animators will usually not time to a reference, but create a desired timing they have in mind, which had to be introduced in order to have a good quantitative measure for timing precision.

With *Sketching*, the path matching algorithm often resulted in the need to manually edit feature points. This was a task many participants had difficulties with, since the correct total number and relative position along paths is essential for an optimal mapping. Some participants adopted strategies of sketching the path at a smaller scale away from the target curve, and a few did not mimic the trajectory at all but created an abstract path that they matched nearly entirely manually.

When *Scrubbing* the timing, subjects were nearly always confused by the mapping between timeline and feature motion. There was a further problem with interactive feedback as some subjects looked at the timeline when performing the scrub, rather than watching the viewport.

An issue occurring with *Dragimation* was that hand and pen occluded the viewport, a problem inherent to direct input. Furthermore, while the algorithm locally matching input to path seems to be well suited for the smooth arc trajectory of task 2, it could run into problems when confronted with the sharp cusps of the motion path in task 3. While dragging over a sharp cusp, the feature could "get stuck" when the target path motion was only followed lazily or short-cut. It then jumped along the path in unwanted jerks, potentially distorting the desired timing.

Subjective Assessment

The rankings of the three techniques that participants gave provide results clearly in favour of Dragimation (figure 6.10). A Friedman test reveals a highly significant ($p < 0.001$) effect for technique for the qualities precision, ease of use, mental load and overall preference. Pairwise Wilcoxon tests show Dragimation to be ranked significantly higher than Scrubbing regarding precision ($p < 0.001$), ease of use ($p = 0.009$), mental load ($p = 0.001$) and overall preference ($p < 0.001$). They also show Dragimation to be ranked significantly higher than Sketching regarding ease of use ($p < 0.001$), mental load ($p = 0.003$) and overall preference ($p = 0.003$). Again, a Mann-Whitney test comparing experts and novices did not reveal any significant effect for group across all four qualities, thus experts did not significantly diverge from novices in their assessment.

The learnability scores achieved (on a scale from 0 to 20) were 12.6 (SD 5.7) for Sketching, 14.8 (SD 5.0) for Scrubbing, and 17.1 (SD 3.45) for Dragimation (figure 6.11), with a highly significant effect for technique (Friedman test, $p < 0.001$). Pairwise Wilcoxon tests showed Scrubbing to score significantly higher than Sketching ($p = 0.005$) and Dragimation to score significantly higher than Scrubbing ($p = 0.024$).

In the interviews, participants almost equivocally judged all three techniques as very intuitive, easy, and quick to use. While most did not enjoy the manual editing often necessary with Sketching, some appreciated it as a means to control the performance mapping. Subjects complained about the lack of input-output correspondence in Scrubbing, which made it difficult to judge which input would lead to which timing. Dragimation was often cited to be the most intuitive tool. Many participants attributed each technique a usefulness for certain application scenarios, although there was no consensus on which was best for what type of task. When asked for a comparison with keyframing tools, the performance timing approach as such was judged to be less precise than keyframe animation, but more suited to create natural, spontaneous timing. It was also thought to be much faster and less cumbersome than the keyframe-based method, and many participants predicted significant productivity improvements. Many also stated that they could well imagine using such tools for prototyping an animation timing, and tweaking details afterwards with standard keyframe tools.

6.3.4 Analysis

The results show that Dragimation lets animators produce timings equally well as Sketching, and better than Scrubbing. This proves the hypothesis that Dragimation has a positive influence on the objective performance of timing regard-

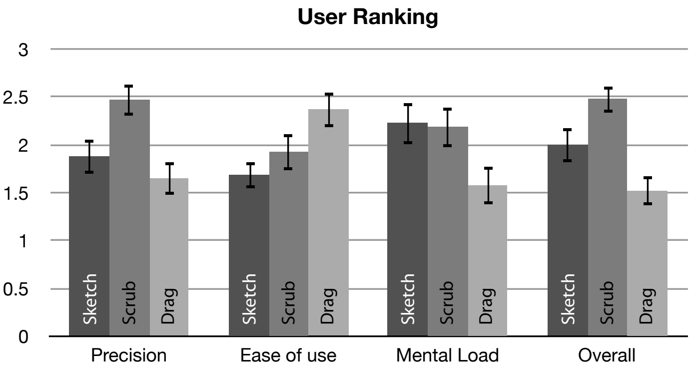


Figure 6.10: Comparison of user ranking means. Precision, mental load, overall: lower is better. Ease of use: higher is better. Error bars indicate 95% confidence interval.

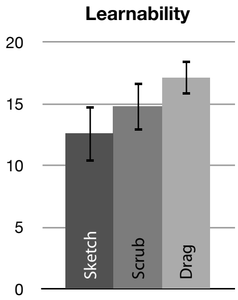


Figure 6.11: Comparison of learnability score means. Error bars indicate 95% confidence interval.

ing the timeline-based Scrubbing technique. This cannot be claimed regarding Dragimation vs. Sketching since their objective performance seems to be about equal. Dragimation was judged to be more precise than Scrubbing, and not less precise than Sketching. All subjects found Dragimation easier to use than Sketching and Scrubbing, with less mental load than the other tools. Dragimation ranked top among all three tools in overall user preference. It was also thought easier to learn than Scrubbing, which in turn was judged easier to learn than Sketching. This proves the second hypothesis that Dragimation has a positive influence on the subjective preference of techniques regarding ease of use, mental load, overall preference and learnability. The interview comments further support these findings. They also show that many participants, including professional animators, could well imagine benefits from using performance timing in animation workflows. Since both objective ability and subjective preference are very important for creative tools, these results can be seen as highly in favour of the direct timing technique.

6.4 Discussion

The contributions to research made in this chapter are summarised in section 6.4.1, before limitations of the treated matter are discussed in section 6.4.2 and ideas for further research are given in section 6.4.3.

6.4.1 Contributions

Dragimation was presented as a new method for performance-based timing of keyframe animations. It was inspired by recent developments in direct manipulation video navigation techniques. The high correspondence between input and output and the interactive feedback of direct manipulation make it a promising candidate for the performance timing task.

A user study with 27 participants of varying experience with animation comparing Dragimation to a sketch-based and a timeline-based technique supports this claim. Dragimation and sketching achieved significantly more precise results than scrubbing in a timing-to-reference task. Dragimation was significantly higher ranked than both other techniques in a subjective assessment regarding ease of use, mental load, overall preference and learnability.

This work intends to make animation timing more accessible by offering an improvement of directness and user satisfaction for timing tools. In a professional setting, animators could use performance-based techniques to develop an initial timing which can then be refined with traditional tools, or others involved in the animation process but untrained in animation tools could use them as a means to better express and communicate their timing ideas. This is supported by the tenor of interviews with professional animators and novices who both expect significant productivity enhancements from using performance timing in an animation workflow. It indicates that such real time tools are a valid option for beginners and more experienced animators alike.

6.4.2 Known Limitations

All three performance timing techniques suffer from the problem of limited timing range. The entire motion must be represented by a concise single-view representation, the fidelity to which this can be displayed and "acted out" is limited by screen and input resolution as much as maximum spatial frequency of human manual motor function. This constrains the length of motion that can be retimed in one session; longer animations must be broken down into manageable segments. Path-based or dragging techniques further need a maximally constant input-motion ratio for optimal use, i.e. the motion path must be mostly parallel to the input plane. While Scrubbing can also be used to time non-spatial phenomena such as colour change, Sketching and Dragimation are limited to space-centric timing. Finally, the performance-based techniques currently cannot produce timings that are too fast for humans to recreate.

Dragimation specifically inherits limitations common to direct-input DMVN techniques. It suffers from occlusion and reach problems typical for direct manipulation on interactive screens. A further problem is temporal ambiguity when a time span is projected onto a single point in image space, when an object stops moving (Karrer et al., 2012). The timing scenario also reveals the effect of physical forces and motor limitations influencing the dragging control. When following the motion path, pen, fingers, hand and arm are subject to real world inertia and other physical restrictions, limiting timing possibilities. While this is also the case for Sketching, its editable mapping provides more flexibility. However, transferring the animator's inertia to the timing can also be seen as a feature. After all, reproducing real world phenomena is what motion capture is all about. In any case, the laws of physics are as much part of the natural panache of humans as experience and intuition. For a comprehensive discussion of direct manipulation time control, the involved issues, and solutions to these see Karrer (2013).

6.4.3 Research Agenda

The following recites solutions to common problems of space-based time control suggested in the literature; with some original thoughts added. Dynamic views could address the problem of limited timing ranges, by reorienting the view based on the range of motion path currently being retimed. The relative flow dragging of Dragicevic et al. (2008) was already used for dragging against moving backgrounds; it remains to be seen how this works for timing. To maintain constant input-output ratio for Sketching, Terra and Metoyer (2007) already suggest finding the optimal path view, e.g. by fitting the optimal view plane to the path; another solution would be to formulate an optimisation problem with the energy being the ratio of 2D and actual 3D arc length[3]. A different approach is to eliminate the problem of varying ratio by using 3D input devices and stereoscopic projection[4]. Multiple views from different angles could find partially best views, which could then be stitched together in a compound multi-view display, allowing the timing of longer animations. Non-spatial phenomena can be timed by making them controllable through spatial handles, while linearly scaling recording speed or even more complex transfer functions could allow timing of speeds that humans cannot achieve (cf. Terra and Metoyer, 2007).

Future work could also address issues specific to Dragimation and DMVN, and some recent work already has taken this up. The problem of cusps in the curve snagging the dragged feature can be counteracted by either improving the proximity metric, or smoothing the curve, e.g. with a Gaussian filter. Using further input DOF, such as contact pressure, might allow interactive control of

---

[3] This was implemented and compared to plane fitting by Nicholas Steenbergen in his bachelor's thesis *Automatic Camera Placement to View Three-Dimensional Curves* (2011).

[4] Investigated by Florian Biermann in his bachelor's thesis *Timing Techniques for Three Degree-of-Freedom Input Devices* (2011).

dragging metric parameters. Interesting extensions of the dragging technique could also be simulation of physical forces, so that features could not only be dragged but flicked or pushed "through time". A solution to problems with occlusion and reach is to provide two duplicate views of the motion path, one for control and the other mainly for visual feedback, at the cost of directness. Karrer et al. (2012) show how conceptually separating control from representation can address temporal ambiguities with *Loops* and *Embedded Timelines* for DMVN.

Non-spatial input in the form of trigger control has been suggested in prior work (section 2.3.3). Using discrete control requires less mental effort, which could be of advantage for interactive timing of secondary phenomena to a given primary motion. By only controlling aspects such as beat, tempo and momentum the artist can concentrate better on observing and anticipating motion to which he is synching. Future work could look into how discrete control points can be made out in motion, either content-based such as the salient feature detection of Terra and Metoyer (2004), or with regular temporal grids.

If the time mapping allows going forwards and backwards in time, instead of retiming existing keyframes one could just capture the traversal of the motion path entirely independent of the prior timing. This is essentially a form of constrained motion capture that has more in common with spatial key framing (Igarashi et al., 2005b) than retiming. For instance, this could be used for efficient animation of cyclic motions such as a swinging pendulum.

The timing scenarios so far only looked at isolated characters or features. A common task is to time a certain motion relative to others, e.g. to synchronise two characters (Kim et al., 2009). Future work should look into partial timing, e.g. relative to constant playback of the rest of the scene (cf. Shah and Narayanan, 2011) or even other parts of the same feature set.

Chapter 7

Discussion

Previous chapters analysed issues of the state of the art in computer animation interfaces and developed guidelines for next generation computer animation tools. These guidelines were applied in two case examples, a surface-based animation system and a technique for motion timing. The contributions to research made in the course were discussed at the end of each chapter together with limitations of the approaches taken. However, certain issues are relevant to the whole work and should be treated independently of individual steps. This chapter summarises the individual contributions of each chapter (section 7.1) and engages points of discussion that have not yet been addressed but are important to the subject of this thesis. It covers the questions of user groups targeted by direct animation (section 7.2) and which areas of application could benefit (section 7.3). Lastly, trade-offs in computer animation are discussed and how these tie in with the work at hand (section 7.4).

7.1 Contributions

This work counts seven major contributions to research. The presentation of related work in chapter 2 offered a *comprehensive discussion of the state of the art* of motion design tools. The *design space of interfaces for time-based visual media* in chapter 3 facilitated analysis, and included a *taxonomy of space-time interactions* that allows better characterisation of the role of real and virtual space and time. In chapter 4 this was used together with a framework from the literature for an *analysis of interaction models in computer animation*, in order to develop *guidelines for direct animation interfaces*. These guidelines were applied in the design, implementation and evaluation of a *surface-based direct animation system* and a *direct animation technique for motion timing* in chapters 5 and 6. The individual contributions and their potential impact are summarised in the following.

7.1.1 Comprehensive Discussion of the State of the Art

Research on animation tools is spread over various disciplines with different perspectives on the problems at hand. The review of related work on motion design interfaces spanned computer graphics, HCI, artificial intelligence, tangible computing and entertainment computing literature. To the knowledge of the author, this is the most comprehensive collection of the state of the art of animation interfaces so far. The discussion was normalised by maintaining a single perspective on the interaction aspects. Research is always shaped by views, opinions and terminology of the respective discipline, which can influence everything from problem definition to approach, analysis and presentation. A consistent view was maintained, making the benefits and drawbacks for the user more evident and making individual contributions comparable. While this was straightforward for some cases it required a closer look at the actual interaction mechanisms at work in others.

7.1.2 Design Space of Interfaces for Time-based Visual Media

Currently, HCI researchers and interaction designers lack the means for a structured investigation of interactions with time-based visual media. A design space was proposed to cover seven dimensions that characterise the most important aspects of such interfaces from conceptual down to technical levels: task, metaphor, directness, integrality, correspondence, manuality and space-time. Where possible, it is based on design issues commonly identified and discussed in the HCI literature. While it was developed for examining motion design interfaces, it can serve purposes beyond this as it covers interactions with time-based visual media in general.

7.1.3 Taxonomy of Space-Time Interactions

The space-time dimension of the design space could not rely on much prior work, since the role of time in interacting with time-based phenomena has hardly been investigated. This led to a new taxonomy for space-time interactions with spatiotemporal media. It is the first structured approach of describing how user and medium space and dynamics relate in human-medium interaction. It can act as a means for classifying concepts in viewing and editing controls for videos, animation, simulation, and games. The classification based on the involved components of real and virtual space and time can serve as a new tool for research and design: by categorising interactions in this way, one can identify mutualities and differences of current techniques and identify which categories of controls are more commonly employed than others. It can potentially point out new alleys and cause new associations, like thinking about how space and time relate to qualities such as liveness, naturalness or precision of a mode of interaction.

7.1.4 Analysis of Interaction Models in Computer Animation

Investigations of novel animation techniques lack the analysis tools to more comprehensively define interaction problems; this is partly owed to them not being connected to examinations of similar trends in HCI. The design space analysis of existing animation techniques facilitated assessing attributes and qualities of common methods and how they relate at a glance. It allowed squaring their characteristics with qualities that the post-WIMP movement strives for. It was shown that while embodied interfaces offer high engagement and thus cognitive directness, manipulation interfaces still provide the best general-use metaphor. The RBI framework of Jacob et al. (2008) is an analysis tool specifically created for post-WIMP generation interfaces. It was illustrated that all four RBI trends can be made out in recent motion design interfaces.

7.1.5 Guidelines for Direct Animation Interfaces

While many works aim to create instruments for more natural and accessible animation, they lack a common foundation, or guidelines. The concept of direct animation interfaces was coined for guiding research, development and design in this application area. Using the notion of direct interaction was justified with its implication in direct-input devices and direct manipulation metaphors as well as its association with principles of reality-based interaction. Based on prior analysis, design guidelines for direct animation were developed for use in designing post-WIMP animation interfaces.

7.1.6 Surface-based Direct Animation System

Interactive surfaces as high-bandwidth direct input devices have a high potential for motion control, but have so far hardly been considered for animation. A design approach for a direct animation system on interactive surfaces was developed and validated by a proof-of-concept prototype. This brought together considerations of surface-based mappings for motion capture and view control as well as strategies for supporting bimanual input. A user study verified the design approach, with participants showing largely positive reactions to the prototype. The clear majority of users employed both hands, supporting the design for bimanual operation. Next to integrating existing techniques and applying prior knowledge in the course of the design process, new techniques were devised to deal with inherent problems in multitrack motion recording on direct-touch devices.

7.1.7 Direct Animation Technique for Motion Timing

Motion timing is an important task in animation, but existing techniques require users trained in the abstractions of the interface. Dragimation was presented as a new method for direct timing of animations. The close spatial input-output correspondence and the interactive feedback of direct manipulation make it a promising candidate for performance timing. A user study comparing Dragimation to a sketching and a scrubbing technique supported this claim. Dragimation and sketching achieved significantly more precise results than scrubbing in a timing-to-reference task. Dragimation was significantly higher ranked than both other techniques in a subjective assessment regarding ease of use, mental load, overall preference and learnability.

7.2 Target Users

This thesis propagates a user-centric approach to analysing and designing animation tools. In the process, general cognitive and manual abilities of humans were considered. Striving for higher usability by applying knowledge on physiological and psychological human factors is the foundation of HCI, but has played little role in computer animation so far. In this context it is important to look at the practitioners using motion design software, their skills and goals and the derived requirements.

Animation is an art and a craft. An animator is a trained expert as well as a talented artist. He must bring an eye for detail, patience, and most importantly skill to his job. Yet the investment in training often decreases interest in innovation. In the experience of the author, working with expert animators is characterised by their scepticism towards alternative methods and tools, and statements of the type "current techniques have proven to work very well" are common. This is not surprising, since these people have invested years in perfecting a certain method. This inertia has also been observed by Davis and Landay (2004), who discovered in interviews with animators that some use regular animation tools even for the prototyping stage of animation. Davis and Landay point out that more rough but faster tools might be more appropriate at early stages of design and could increase efficiency. Direct animation takes this one step further with the vision of fast *and* precise tools for every stage of the design process. Its ultimate goal is to eliminate the trade-off between precision and efficiency (section 7.4).

But expert users lie at the end of a whole spectrum of user proficiency. There are plenty intermediate users and beginners that are either still in education, or their practice simply does not require or allow a higher level of accomplishment. Animation tools cannot enable beginners to create stunning motion designs without significantly constraining creativity (see discussion of trade-offs in section 7.4). However, they can make the learning curve less steep. Experienced animators have memorised abstractions that allow them to operate complex software, such as how certain dynamics transfer into numbers of frames. Novices or casual users lack such skills, yet there are many good reasons to better support this group. Animations and consequently animators are in high demand, and the profession cannot afford a high entry threshold for motivated novices. Furthermore, animation production chains are highly specialised and require users of varying skill and specialisation to collaborate. These workflows have a lot to gain if all involved—from storyboard artist and character designers to riggers, lighters and animators—possess a basic level of animation proficiency. Svensson et al. (2008) report on a discussion with professional animators who observed that direct animation tools could create a bridge between (frame-based) computer animation and traditional puppeteers, which could make it easier to find new practitioners and *democratise* the profession with regard to the competencies required (see also section 7.3).

Direct animation interfaces should incorporate everyone from beginners to experienced professionals, by being easy to learn, but hard to master. This view resonates with voices in the community that, rather than making systems easy to use, intend to *accelerate the process whereby novices perform like experts* (Kabbash et al., 1994), and *make beginners behave like professionals* by letting them feel like *naturals* (Wigdor and Wixon, 2011).

7.3 Applying Direct Animation

In order to gain a better sense of its potential impact, it is helpful to deliberate how direct animation can be applied in practice. The following projection starts at the lowest level, the design tasks, considers stages in animation production, and ends with a survey of application areas in authored and live media.

7.3.1 Motion Design Tasks

Character animation is possibly the most ostensive task in animation, and the one that can potentially benefit the most from direct animation. Performance animation is already heavily used in creating motion for virtual humans and human-like creatures. Direct animation interfaces can go beyond literal mappings for virtual acting and provide puppet-like control for non-humanoid and general anthropomorphic targets. Compared to indirect, offline animation these can offer more spontaneity and thus character and believability (cf. section 7.4).

Svensson et al. (2008) cite animators suggesting to use performance-based, direct approaches for non-character animation as well. Sets and props that feel alive are as essential to storytelling as believable characters. In many contexts such as advertisement, information visualisation, or communication and education, animating non-anthropomorphic targets can even be the main task (see also section 7.3.2). Often this will require directing a simulation rather than controlling all degrees of freedom. Making simulations art-directable is the goal of recent research, e.g. for designing simulations of rigid bodies (Popović et al., 2000) or deformable materials (Schumacher et al., 2012). However, the cited works only target frame-based animation tools. Applying direct animation would foster direct real time control of physical simulations, which is so far only considered for character puppeteering (section 2.3).

7.3.2 Production Stages

In a typical animation production workflow, long before an artist details any character behaviour, first design decisions are made with the help of coarse motion designs that block out the animation. Such animatics, which are essentially moving storyboards, are an important step in many animators' work process (Davis and Landay, 2004). Since this stage of design is more about finding the right sequence and timing of events rather than the exact definition of motion it has different requirements for degree of control and precision. Rather than using standard animation tools, this essential stage of production can benefit from rougher, more informal, and more direct design tools, which as a rule are more efficient (Davis and Landay, 2004).

Workflows in big film, TV and games productions are highly specialised and separated (Baecker, 1969; PIXAR, 2010). Many professions apart from actual animators are involved, such as storyboard artists, character and set designers, audio artists, technical staff, and directors. More accessible animation tools may open up new possibilities: storyboard artists could visualise motions that directors could edit before the specialist goes on to do the finishing touches. Everybody involved would benefit from this "democratisation" of proficiency, and workflow efficiency is likely to increase. Rather than compromising their sovereignty this could improve working conditions of motion designers: they would benefit by getting more ideas and by having more time for details and nuances rather than basic scene layouts.

The final stage of motion design, in which motions are refined and details added, has the most uncertain appreciation for direct animation. In the experience of the author, animators trained in frame-based methods are more interested in precise control than spontaneity or efficiency. However, they are obviously biased by their discipline. It is the central argument of this thesis that even for refined animation, direct, performance-based approaches can be appropriate, given the right control interface.

7.3.3 Applications in Authored Media

The principal and "home" application field of animation is video entertainment in film, television, and advertisement. It is here that animation historically originated—to amuse the masses. These industries with their large budgets and complex production pipelines potentially have a lot to gain from direct animation systems, as discussed previously. Yet there are many other areas of application for authoring tools for dynamic content.

Interactive digital entertainment has outgrown the market for any other mass medium. Video games are rich in visual dynamic content, typically featuring many pre-authored animations that are activated by player input or game mechanics. Interactive environments have different requirements on quantity and quality of animations than do linear media.

Animation is also indispensable for portraying dynamic processes in information visualisation. While dynamics are often simulated in science and engineering, direct animation can help to highlight or annotate dynamic data. Interactive control of simulations can aid exploring potential or hypothetical scenarios that are not easily calculated. Visualisation is also essential in industry, commerce, and planning. Here, direct animation provides a comparatively cheap and powerful way to create displays of products, locations, or processes. The democratisation of proficiency could give part-time animators and non-experts such as architects, industrial designers, or marketing personnel more control over the dynamics of visual communication.

A further field is education, where direct tools can support creators of teaching materials to provide illustrative animated content in their products. Teachers and tutors themselves could be empowered to create small dynamic visualisations for their classes. With the proliferation of e-books and digital readers, the demand for animated content that utilises the power of digital display technology is already there and will potentially grow further.

Direct animation can even serve as the concept underlying general-purpose presentation tools. These already feature highly indirect animation functionality or provide predefined animations, often resulting in bad or stale animations. The general promise of animation for communication is pointed out by Davis and Landay (2004): It can represent dynamic concepts and make information more attractive and engaging. "Thus, it can be argued that the ability to create animation can make anyone a better communicator."

7.3.4 Applications in Live Media

Direct animation embraces real time online capture of input for direct control of dynamics. This places the act of motion design closer to a performance than any traditional method: the animator essentially becomes a performer. As such the act of creating a persistent artefact also obtains dramaturgical aesthetics in itself. This increases the overlap to more ephemeral art forms and introduces the notion of a fluent spectrum from live animation to the performing arts.

A similar spectrum can be observed in interfaces for real time games. Such game controls are essentially constrained puppetry interfaces for an avatar or even a whole game world. They are used beyond mere gameplay to create choreographies or even stories. The potential of *machinima*—movies that are virtually produced within game engines—for cheaper production in the animation industry has already been spotted (section 2.5.3). The spread of motion-based game controls further increases the relation to performance animation controls.

In other areas direct animation interfaces can also contribute to blurring the line between controls for authored media and live presentations. Animating dynamic phenomena live in front of audiences, whether they be scientists or primary school students, would have significant didactic benefits. Viewers could witness the creation live in order to better understand the process and then view the result afterwards for reference, possibly with a playback of individual animation stages or layers. Since the presenter is also the author, he can better relate to the content than when playing back canned material. Direct animation tools thus can potentially make a sales pitch, virtual product demonstration or school lesson as engaging as a musical performance in which a solo musician builds up a multi-instrument piece layer by layer with an audio looper.

7.4 Trade-offs in Computer Animation

Having to create entire worlds along with the characters that populate them is extremely laborious. Character animation in particular requires the specification of dozens of values for defining a single pose. This conundrum is known in the literature as the *degrees of freedom problem* (Zeltzer, 1985). As leading industry animators put it, the challenge of the computer animation medium is that it "contains nothing, down to the smallest detail, that you do not create yourself. You get nothing for free" (PIXAR, 2010). Ultimate control comes at the price of having to define all and everything. This leads to a significant efficiency problem, with high ratios of production to animation time.

Various attempts have been made and are being pursued in research and development to address this issue by submitting control in favour of other advantages. While the main benefit is usually increased efficiency, other factors such as increased realism, simpler interfaces or a looseness of motion also play a role. In the following such *trade-offs* between control on the one hand and automation, simplicity, and spontaneity on the other are inspected.

7.4.1 Control versus Automation

Certain animation problems exceed the abilities of manual specification. Examples are large-scale, complex processes such as particle movements of water and smoke, crowd behaviour, and complex physical interactions of all kinds.

The goal of *automation* is to increase efficiency in such processes or make them feasible in the first place. With automated methods, the animator hands over control to scripted procedures or simulation rules. The downside is that tweaking simulation parameters makes control highly indirect.

The trade-off lies in finding the right balance between automated complexity and expressive control (cf. Zeltzer, 1985). Hybrid approaches can aid in finding this balance by combining the best of automated and manual control, as exemplified by art-directable simulations (e.g. Popović et al., 2000; Schumacher et al., 2012) or partial simulation in real time animation (e.g. Oore et al., 2002b).

7.4.2 Control versus Simplicity

Controlling many degrees of freedom tends to require complex interfaces. Examples are the multitude of instruments used in frame animation, or obtrusive capture hardware and coordination overhead in performance animation. This complex interaction creates inefficiency and a high entry threshold.

Increasing tool *simplicity* can be achieved by limiting the control space or constraining it based on pre-defined models. For instance, animation functionality of slide-based presentation software gains its simplicity by supporting only a limited range of template motion designs (Davis and Landay, 2004). Constraints can use mathematical models, examples, heuristics or the designer's intuition or experience. They are used to simplify general manipulation (section 2.1) as well as performance controls (section 2.3).

The trade-off lies in finding the amount of control desired and the user's level of training and ambition to learn the interface, as well as the expressivity required for the animation task. Tools for casual users or limited scenarios are often more accessible by providing very constrained control, while professional tools feature complex but powerful interfaces.

7.4.3 Control versus Spontaneity

Time-independent control over motion and dynamics also eliminates spontaneity in the creative process. In frame-based animation, if every moment in animation time is meticulously designed over several hours or days, none of the spontaneity and wit of its creators will immediately reflect in the result. Rather, their expressiveness is mediated by the instruments of control. In other media, artists will often be inspired by the moment or let their surroundings immediately influence their work. In strongly mediated animation methods, there will be no such spur-of-the-moment: "There are never any lucky accidents in the computer, only hard-won victories." (PIXAR, 2010).

Performance-based approaches hand over control to the vagaries of the situation and to the performer's intuition for spontaneous decisions. Real time control brings both efficiency and a quality of looseness. For instance, the Jim Henson company uses digital puppetry in the production of children's television shows, giving them the look of 3D animation, but also "the looseness and the fun-ness of a performed medium" (Jurgensen, 2008). A certain emergent and spontaneous use of control can also benefit general interactions, as physics-based mappings for object manipulation demonstrate (cf. section 2.1).

The trade-off thus lies in submitting a certain precision and time-independence in motion editing and embracing the creativity of the here-and-now with all its quirks and coincidences. The downsides of time-dependence are that mistakes cannot be easily edited, but rather must be redone entirely as a new capture take. It is up to software to address these issues, the work presented in this thesis provides a first step in this direction.

7.5 Conclusion

The dominance of indirect animation methods is fading. Until the invention of moving imagery, animation was always live, acted out by performers. Film brought conservation of live performances. Frame animation brought entirely new ways of controlling what was shown, but also introduced a distance between the artist and the medium. For a long time this was the best means for visual storytelling free of the constraints posed by real actors, props and sets. The advancement of modern technology has changed this. Motion capture has reintroduced the most traditional form of animation—animation by performance—and exposed it to the vast possibilities of the digital age.

This thesis explored these possibilities by establishing direct animation interfaces. It argued that connecting research on computer animation to HCI concepts and methods could inform the design of a new generation of motion design tools. In order to explore the validity of this hypothesis, goals were formulated to address issues of interface use and design in this area. The summary of contributions shows that these goals could be met. The comprehensive discussion of the state of the art, the design space and the taxonomy of space-time mappings established the HCI perspective and embedded animation methods in this context. This in turn aided relating previously disconnected trends in computer graphics and reality-based interaction that have a lot in common regarding certain qualities of tool use. Using the design space and the RBI framework, the resulting insights were distilled into guidelines for direct animation interfaces that minimise specific issues of abstraction, inefficiency, and limitations of contemporary tools. The presentation of two point designs—a direct animation system and a direct motion timing technique—demonstrated that this foundation is a formidable basis for designing novel systems and techniques for more direct, efficient and flexible motion design tools. Studies gave insight on the impact of this approach on aspects of use such as directness, efficiency and user satisfaction. The final discussion related these solutions to potential users and applications, and paid tribute to the fact that no solution can solve every problem and individual design choices must always be made. It illustrated the potential of direct animation in many areas, showing that this work has taken first steps on a path leading to more powerful, accessible, and expressive animation of tomorrow.

Bibliography

Anand Agarawala and Ravin Balakrishnan. Keepin' it real: pushing the desktop metaphor with physics, piles and the pen. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '06, pages 1283–1292, New York, NY, USA, 2006. ACM.

Ronald M. Baecker. Picture-driven animation. In *Proceedings of the Spring Joint Computer Conference*, AFIPS '69 (Spring), pages 273–288, New York, NY, USA, 1969. ACM.

Jean-Francis Balaguer and Enrico Gobbetti. Sketching 3D animations. *Computer Graphics Forum*, 14(3):241–258, 1995.

Ravin Balakrishnan and Ken Hinckley. The role of kinesthetic reference frames in two-handed input performance. In *Proceedings of the Annual ACM Symposium on User Interface Software and Technology*, UIST '99, pages 171–178, New York, NY, USA, 1999. ACM.

Ravin Balakrishnan and Ken Hinckley. Symmetric bimanual interaction. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, CHI '00, pages 33–40, New York, NY, USA, 2000. ACM.

Ravin Balakrishnan and Gordon Kurtenbach. Exploring bimanual camera control and object manipulation in 3D graphics interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '99, pages 56–62, New York, NY, USA, 1999. ACM.

Connelly Barnes, David E. Jacobs, Jason Sanders, Dan B. Goldman, Szymon Rusinkiewicz, Adam Finkelstein, and Maneesh Agrawala. Video puppetry: a performative interface for cutout animation. *ACM Transactions on Graphics*, 27(5):124:1–124:9, December 2008.

Michel Beaudouin-Lafon. Instrumental interaction: an interaction model for designing post-WIMP user interfaces. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, CHI '00, pages 446–453, New York, NY, USA, 2000. ACM.

Hrvoje Benko, Andrew D. Wilson, and Patrick Baudisch. Precise selection techniques for Multi-Touch screens. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '06, pages 1263–1272, New York, NY, USA, April 2006. ACM.

François Bérard, Jessica Ip, Mitchel Benovoy, Dalia El-Shimy, Jeffrey R. Blum, and Jeremy R. Cooperstock. Did " minority report" get it wrong? superiority of the mouse over 3D input devices in a 3D placement task. In Tom Gross, Jan Gulliksen, Paula Kotzé, Lars Oestreicher, Philippe Palanque, RaquelOliveira Prates, and Marco Winckler, editors, *Proceedings Human-Computer Interaction*, INTERACT '09, pages 400–414, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.

Bobby Bodenheimer, Chuck Rose, Seth Rosenthal, and John Pella. The process of motion capture: Dealing with the data. In *Proceedings of the Eurographics Workshop on Computer Animation and Simulation*, Eurographics, pages 3–18. Springer Vienna, 1997.

Doug A. Bowman, David Koller, and Larry F. Hodges. Travel in immersive virtual environments: An evaluation of viewpoint motion control techniques. In *Virtual Reality Annual International Symposium*, pages 45–52, Los Alamitos, CA, USA, March 1997. IEEE Computer Society.

Doug A. Bowman, Donald B. Johnson, and Larry F. Hodges. Testbed evaluation of virtual environment interaction techniques. In *Proceedings of the ACM symposium on Virtual reality software and technology*, VRST '99, pages 26–33, New York, NY, USA, 1999. ACM.

Doug A. Bowman, Ernst Kruijff, Joseph J. LaViola, and Ivan Poupyrev. *3D User Interfaces: Theory and Practice*. Addison-Wesley, 2004.

Christoph Brachmann and Rainer Malaka. Keyframe-less integration of semantic information in a video player interface. In *Proceedings of the Seventh European Conference on Interactive Television*, EuroITV '09, pages 137–140, New York, NY, USA, 2009. ACM.

Brightside Games. *Zeit2*. Ubisoft, 2011.

William Buxton and Brad A. Myers. A study in two-handed input. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '86, pages 321–326, New York, NY, USA, 1986. ACM.

Gordon Cameron, Andre Bustanoby, Ken Cope, Steph Greenberg, Craig Hayes, and Olivier Ozoux. Motion capture and CG character animation (panel). In *Proceedings of the Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '97, pages 442–445, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co.

Xiang Cao, Andrew D. Wilson, Ravin Balakrishnan, Ken Hinckley, and Scott E. Hudson. ShapeTouch: Leveraging contact shape on interactive surfaces. In *Proceedings of International Workshop on Horizontal Interactive Human Computer Systems*, TABLETOP '08, pages 129–136, Los Alamitos, CA, USA, 2008. IEEE Computer Society.

Stuart K. Card. Human factors and artificial intelligence. In P. A. Hancock and M. H. Chignell, editors, *Intelligent Interfaces: Theory, Research and Design*, pages 270–284. Elsevier Science Publishers B.V., Amsterdam, 1989.

Stuart K. Card and Jock D. Mackinlay. The structure of the information visualization design space. In *Proceedings of IEEE Symposium on Information Visualization*, pages 92–99, Los Alamitos, CA, USA, October 1997. IEEE Computer Society.

Stuart K. Card, Jock D. Mackinlay, and George G. Robertson. The design space of input ddevices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '90, pages 117–124, New York, NY, USA, 1990. ACM.

Stuart K. Card, Jock D. Mackinlay, and George G. Robertson. A morphological analysis of the design space of input devices. *ACM Transactions on Information Systems*, 9(2):99–122, April 1991.

Daniel Casasanto and Lera Boroditsky. Time in the mind: Using space to think about time. *Cognition*, 106(2):579–593, February 2008.

Jinxiang Chai and Jessica K. Hodgins. Performance animation from low-dimensional control signals. *ACM Transactions on Graphics*, 24(3):686–696, July 2005.

Kwang-jin Choi and Hyeong-seok Ko. On-line motion retargetting. *Journal of Visualization and Computer Animation*, 11:223–235, 1999.

Kensy Cooperrider and Rafael Núñez. Doing time: speech, gesture, and the conceptualization of time. Technical Report 3, Center for Research in Language, University of California, December 2007.

H. A. Cunningham and R. B. Welch. Multiple concurrent visual-motor mappings: implications for models of adaptation. *Journal of experimental psychology. Human perception and performance*, 20(5):987–999, October 1994.

Lawrence D. Cutler, Bernd Fröhlich, and Pat Hanrahan. Two-handed direct manipulation on the responsive workbench. In *Proceedings of the Symposium on Interactive 3D Graphics*, I3D '97, pages 107–114, New York, NY, USA, 1997. ACM.

James Davis, Maneesh Agrawala, Erika Chuang, Zoran Popović, and David Salesin. A sketching interface for articulated figure animation. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '03, pages 320–328, Aire-la-Ville, Switzerland, 2003. Eurographics Association.

Richard C. Davis and James A. Landay. Informal animation sketching: Requirements and design. Technical report, Group for User Interface Research, Computer Science Division, 2004.

Richard C. Davis, Brien Colwell, and James A. Landay. K-sketch: a 'kinetic' sketch pad for novice animators. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, pages 413–422, New York, NY, USA, 2008. ACM.

Frank R. Dellario. The future of machinima as a professional animation resource and its growth as Real-Time animation in virtual worlds. *Journal of Visual Culture*, 10(1):89–92, April 2011.

John D. Dionisio and Alfonso F. Cárdenas. A unified data model for representing multimedia, timeline, andsimulation data. *IEEE Transactions on Knowledge and Data Engineering*, 10(5):746–767, September 1998.

Mira Dontcheva, Gary Yngve, and Zoran Popović. Layered acting for character animation. *ACM Transactions on Graphics*, 22(3):409–416, July 2003.

Tanja Döring, Axel Sylvester, and Albrecht Schmidt. A design space for ephemeral user interfaces. In *Proceedings of the 7th International Conference on Tangible, Embedded and Embodied Interaction*, TEI '13, pages 75–82, New York, NY, USA, 2013. ACM.

Paul Dourish. *Where the Action Is: The Foundations of Embodied Interaction*. MIT Press, Cambridge, MA, USA, 2001.

Pierre Dragicevic, Gonzalo Ramos, Jacobo Bibliowitcz, Derek Nowrouzezahrai, Ravin Balakrishnan, and Karan Singh. Video browsing by direct manipulation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, pages 237–246, New York, NY, USA, 2008. ACM.

Jörg Edelmann, Andreas Schilling, and Sven Fleck. The DabR - a multitouch system for intuitive 3D scene navigation. In *3DTV Conference: The True Vision - Capture, Transmission and Display of 3D Video, 2009*, pages 1–4, Los Alamitos, CA, USA, May 2009. IEEE Computer Society.

Julien Epps, Serge Lichman, and Mike Wu. Study of hand shape use in tabletop gesture interaction. In *CHI '06 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '06, pages 748–753, New York, NY, USA, 2006. ACM.

Chris Esposito, W. Bradford Paley, and JueyChong Ong. Of mice and monkeys: a specialized input device for virtual body animation. In *Proceedings of the Symposium on Interactive 3D Graphics*, I3D '95, New York, NY, USA, 1995. ACM.

Vyvyan Evans. *The structure of time: language, meaning and temporal cognition.* John Benjamins, 2004.

James D. Foley, Andries van Dam, Steven K. Feiner, and John F. Hughes. *Computer Graphics - Principles and Practice*. Addison-Wesley, 1996.

Clifton Forlines, Daniel Vogel, and Ravin Balakrishnan. HybridPointing: fluid switching between absolute and relative pointing with a direct input device. In *Proceedings of the Annual ACM Symposium on User Interface Software and Technology*, UIST '06, pages 211–220, New York, NY, USA, 2006. ACM.

Clifton Forlines, Daniel Wigdor, Chia Shen, and Ravin Balakrishnan. Direct-touch vs. mouse input for tabletop displays. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '07, pages 647–656, New York, NY, USA, 2007. ACM.

Bernd Fröhlich, Henrik Tramberend, Andrew Beers, Maneesh Agrawala, and David Baraff. Physically-based manipulation on the responsive workbench. In *Proceedings of IEEE Virtual Reality*, pages 5–11, Los Alamitos, CA, USA, 2000. IEEE Computer Society.

David M. Frohlich. The design space of interfaces. In Lars Kjelldahl, editor, *Proceedings of the Eurographics Workshop on Multimedia*, EurographicSeminars, pages 53–69. Springer Berlin Heidelberg, 1992.

David M. Frohlich. Direct manipulation and other lessons. In Martin G. Helander, Thomas K. Landauer, and Prasad V. Prabhu, editors, *Handbook of Human-Computer Interaction*, pages 463–488. North-Holland, Elsevier, 1997.

Chi W. Fu, Wooi B. Goh, and Junxiang A. Ng. Multi-touch techniques for exploring large-scale 3D astrophysical simulations. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '10, pages 2213–2222, New York, NY, USA, 2010. ACM.

Tinsley A. Galyean. Guided navigation of virtual environments. In *Proceedings of the 1995 symposium on Interactive 3D graphics*, I3D '95, pages 103–ff., New York, NY, USA, 1995. ACM.

Wendell R. Garner. *The processing of information and structure.* Lawrence Erlbaum, Oxford, England, 1974.

Gildfind, M. A. Gigante, and G. Al-Qaimari. Evolving performance control systems for digital puppetry. *The Journal of Visualization and Computer Animation*, pages 169–183, September 2000.

Michael Gleicher. Motion editing with spacetime constraints. In *Proceedings of the Symposium on Interactive 3D Graphics*, I3D '97, New York, NY, USA, 1997. ACM.

Michael Gleicher. Retargetting motion to new characters. In *Proceedings of the Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '98, pages 33–42, New York, NY, USA, 1998. ACM.

Michael Gleicher. Animation from observation: Motion capture and motion editing. *Computer Graphics*, 33(4):51–54, November 1999.

Michael Gleicher. Motion path editing. In *Proceedings of the 2001 symposium on Interactive 3D graphics*, I3D '01, pages 195–202, New York, NY, USA, 2001. ACM.

Michael Gleicher and Andrew Witkin. Through-the-lens camera control. *Computer Graphics*, 26(2):331–340, July 1992.

Enrico Gobbetti and Jean-Francis Balaguer. An integrated environment to visually construct 3D animations. In *Proceedings of the Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '95, pages 395–398, New York, NY, USA, 1995. ACM.

Dan B. Goldman, Chris Gonterman, Brian Curless, David Salesin, and Steven M. Seitz. Video object annotation, navigation, and composition. In *Proceedings of the Annual ACM Symposium on User Interface Software and Technology*, UIST '08, pages 3–12, New York, NY, USA, 2008. ACM.

Anthony Grafton and Daniel Rosenberg. *Cartographies of Time*. Princeton Architectural Press, 2010.

Tovi Grossman and David Wigdor. Going deeper: a taxonomy of 3D on the tabletop. In *Proceedings of International Workshop on Horizontal Interactive Human Computer Systems*, TABLETOP '07, pages 137–144. IEEE Computer Society, 2007.

Yves Guiard. Asymmetric division of labor in human skilled bimanual action: the kinematic chain as a model. *Journal of motor behavior*, 19(4):486–517, December 1987.

Benjamin Hagedorn and Jürgen Döllner. Sketch-Based navigation in 3D virtual environments. In *Proceedings of the International Symposium on Smart Graphics*, pages 239–246, Berlin, Heidelberg, 2008. Springer-Verlag.

Marc S. Hancock, Frédéric D. Vernier, Daniel Wigdor, Sheelagh Carpendale, and Chia Shen. Rotation and translation mechanisms for tabletop interaction. In *Proceedings of International Workshop on Horizontal Interactive Human Computer Systems*, TABLETOP '06, Los Alamitos, CA, USA, January 2006. IEEE Computer Society.

Mark Hancock, Sheelagh Carpendale, and Andy Cockburn. Shallow-depth 3d interaction: design and evaluation of one-, two- and three-touch techniques. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '07, pages 1147–1156, New York, NY, USA, 2007. ACM.

Mark Hancock, Thomas T. Cate, and Sheelagh Carpendale. Sticky tools: Full 6DOF Force-Based interaction for Multi-Touch tables. In *Proceedings of Interactive Tabletops and Surfaces 2009*, New York, NY, USA, 2009. ACM.

Marc Herrlich. *3D-Modellierung mit interaktiven Oberflächen*. PhD thesis, University of Bremen, Germany, 2013.

Marc Herrlich, Benjamin Walther-Franks, and Rainer Malaka. Integrated rotation and translation for 3D manipulation on Multi-Touch interactive surfaces. In *Proceedings of the International Symposium on Smart Graphics*, pages 146–154, Berlin, Heidelberg, July 2011. Springer-Verlag.

Otmar Hilliges, Shahram Izadi, Andrew D. Wilson, Steve Hodges, Armando G. Mendoza, and Andreas Butz. Interactions in the air: adding further depth to interactive tabletops. In *Proceedings of the Annual ACM Symposium on User Interface Software and Technology*, UIST '09, pages 139–148, New York, NY, USA, 2009. ACM.

Ken Hinckley and Daniel Wigdor. Input technologies and techniques. In Julie A. Jacko, editor, *The Human-Computer Interaction Handbook – Fundamentals, Evolving Technologies and Emerging Applications, Third Edition*, chapter 9. Taylor & Francis, 2012.

Ken Hinckley, Randy Pausch, John C. Goble, and Neal F. Kassell. A survey of design issues in spatial input. In *Proceedings of the Annual ACM Symposium on User Interface Software and Technology*, UIST '94, pages 213–222, New York, NY, USA, 1994. ACM.

Ken Hinckley, Randy Pausch, Dennis Proffitt, and Neal F. Kassell. Two-handed virtual manipulation. *ACM Transactions on Computer-Human Interaction*, 5 (3):260–302, September 1998.

Jessica K. Hodgins. Animating human motion. *Scientific American*, 278(3): 64–69, 1998.

Eugene Hsu, Marco da Silva, and Jovan Popović. Guided time warping for motion editing. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '07, pages 45–52, Aire-la-Ville, Switzerland, 2007. Eurographics Association.

Wolfgang Hürst, Georg Götz, and Philipp Jarvers. Advanced user interfaces for dynamic video browsing. In *Proceedings of the 12th annual ACM international* *conference on Multimedia*, MULTIMEDIA '04, pages 742–743, New York, NY, USA, 2004. ACM.

Edwin L. Hutchins, James D. Hollan, and Donald A. Norman. Direct manipulation interfaces. *Human-Computer Interaction*, 1(4):311–338, December 1985.

Takeo Igarashi, Rieko Kadobayashi, Kenji Mase, and Hidehiko Tanaka. Path drawing for 3D walkthrough. In *Proceedings of the Annual ACM Symposium on User Interface Software and Technology*, UIST '98, pages 173–174, New York, NY, USA, 1998. ACM.

Takeo Igarashi, Tomer Moscovich, and John F. Hughes. As-rigid-as-possible shape manipulation. *ACM Transactions on Graphics*, 24(3):1134–1141, 2005a.

Takeo Igarashi, Tomer Moscovich, and John F. Hughes. Spatial keyframing for performance-driven animation. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '05, pages 107–115, New York, NY, USA, 2005b. ACM.

James N. Ingram, Konrad P. Körding, Ian S. Howard, and Daniel M. Wolpert. The statistics of natural hand movements. *Experimental brain research. Experimentelle Hirnforschung. Expérimentation cérébrale*, 188(2):223–236, June 2008.

Satoru Ishigaki, Timothy White, Victor B. Zordan, and C. Karen Liu. Performance-based control interface for character animation. *ACM Transactions on Graphics*, 28(3):1–8, July 2009.

Hiroshi Ishii and Brygg Ullmer. Tangible bits: towards seamless interfaces between people, bits and atoms. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '97, pages 234–241, New York, NY, USA, 1997. ACM.

Robert J. K. Jacob, Linda E. Sibert, Daniel C. McFarlane, and M. Preston Mullen. Integrality and separability of input devices. *ACM Transactions on Computer-Human Interaction*, 1(1):3–26, March 1994.

Robert J. K. Jacob, Audrey Girouard, Leanne M. Hirshfield, Michael S. Horn, Orit Shaer, Erin T. Solovey, and Jamie Zigelbaum. Reality-based interaction: a framework for post-WIMP interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, pages 201–210, New York, NY, USA, 2008. ACM.

Michael P. Johnson, Andrew D. Wilson, Bruce Blumberg, Christopher Kline, and Aaron Bobick. Sympathetic interfaces: using a plush toy to direct synthetic characters. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '99, pages 152–158, New York, NY, USA, 1999. ACM.

John Jurgensen. From muppets to digital puppets. `http://www.youtube.com/watch?v=GN8WbHomQJg` (last accessed 6th January 2019), August 2008.

Paul Kabbash, William Buxton, and Abigail Sellen. Two-handed input in a compound task. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '94, pages 417–423, New York, NY, USA, 1994. ACM.

Martin Kaltenbrunner, Till Bovermann, Ross Bencina, and Enrico Costanza. TUIO - a protocol for table based tangible user interfaces. In *Proceedings of the 6th International Workshop on Gesture in Human-Computer Interaction and Simulation (GW 2005)*, Vannes, France, 2005.

Thorsten Karrer. *Semantic Navigation in Digital Media*. PhD thesis, RWTH Aachen University, July 2013.

Thorsten Karrer, Malte Weiss, Eric Lee, and Jan Borchers. DRAGON: a direct manipulation interface for frame-accurate in-scene video navigation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, pages 247–250, New York, NY, USA, 2008. ACM.

Thorsten Karrer, Moritz Wittenhagen, and Jan Borchers. DragLocks: handling temporal ambiguities in direct manipulation video navigation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '12, pages 623–626, New York, NY, USA, 2012. ACM.

Matt Kelland, Dave Morris, and Dave Lloyd. *Machinima: Making Animated Movies in 3D Virtual Environments*. Ilex, Lewes, 2005.

Ji-Sun Kim, Denis Gračanin, Krešimir Matković, and Francis Quek. Finger walking in place (FWIP): A traveling technique in virtual environments. In *Proceedings of the International Symposium on Smart Graphics*, pages 58–69, Berlin, Heidelberg, 2008. Springer-Verlag.

Manmyung Kim, Kyunglyul Hyun, Jongmin Kim, and Jehee Lee. Synchronized multi-character motion editing. *ACM Transactions on Graphics*, 28(3):1–9, July 2009.

Don Kimber, Tony Dunnigan, Andreas Girgensohn, Frank Shipman, Thea Turner, and Tao Yang. Trailblazing: Video playback control by direct object manipulation. In *Proceedings of IEEE International Conference on Multimedia and Expo*, pages 1015–1018, Los Alamitos, CA, USA, July 2007. IEEE Computer Society.

Kenrick Kin, Maneesh Agrawala, and Tony DeRose. Determining the benefits of direct-touch, bimanual, and multifinger input on a multitouch workstation. In *Proceedings of Graphics Interface*, GI '09, pages 119–124, Toronto, Ontario, Canada, 2009. Canadian Information Processing Society.

Michael Kipp and Quan Nguyen. Multitouch puppetry: creating coordinated 3D motion for an articulated arm. In *Proceedings of the International Conference on Interactive Tabletops and Surfaces*, ITS '10, pages 147–156, New York, NY, USA, 2010. ACM.

Brian Knep, Craig Hayes, Rick Sayre, and Tom Williams. Dinosaur input device. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '95, pages 304–309, New York, NY, USA, 1995. ACM Press/Addison-Wesley Publishing Co.

Philipp Kolhoff, Jaqueline Preuß, and Jörn Loviscach. Walking with pens. In J. Dingliana and F. Ganovelli, editors, *Eurographics 2005 Short Presentations*, 2005.

Haruki Kouda, Ichiroh Kanaya, and Kosuke Sato. Motion Stroke-A Tablet-Based interface for motion design tool using drawing. In *Proceedings of the 13th International Conference on Human-Computer Interaction. Part II: Novel Interaction Methods and Techniques*, pages 821–829, Berlin, Heidelberg, 2009. Springer-Verlag.

Russell Kruger, Sheelagh Carpendale, Stacey D. Scott, and Saul Greenberg. How people use orientation on tables: comprehension, coordination and communication. In *Proceedings of the International ACM SIGGROUP Conference on Supporting Group Work*, GROUP '03, pages 369–378, New York, NY, USA, 2003. ACM.

Russell Kruger, Sheelagh Carpendale, Stacey D. Scott, and Anthony Tang. Fluid integration of rotation and translation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '05, pages 601–610, New York, NY, USA, 2005. ACM.

Wai C. Lam, Feng Zou, and Taku Komura. Motion editing with data glove. In *Proceedings of the 2004 ACM SIGCHI International Conference on Advances in Computer Entertainment Technology*, ACE '04, pages 337–342, New York, NY, USA, 2004. ACM.

John Lasseter. Principles of traditional animation applied to 3D computer animation. *Computer Graphics*, 21(4):35–44, 1987.

Joseph Laszlo, Michiel van de Panne, and Eugene Fiume. Interactive control for physically-based animation. In *Proceedings of the Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '00, pages 201–208, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co.

Jehee Lee and Sung Y. Shin. A hierarchical approach to interactive motion editing for human-like figures. In *Proceedings of the Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '99, pages 39–48, New York, NY, USA, 1999. ACM Press/Addison-Wesley Publishing Co.

Jehee Lee, Jinxiang Chai, Paul S. A. Reitsma, Jessica K. Hodgins, and Nancy S. Pollard. Interactive control of avatars animated with human motion data. *ACM Transactions on Graphics*, 21(3):491–500, July 2002.

Andrea Leganchuk, Shumin Zhai, and William Buxton. Manual and cognitive benefits of two-handed input: an experimental study. *ACM Transactions on Computer-Human Interaction*, 5(4):326–359, December 1998.

James R. Lewis and Jeff Sauro. The factor structure of the system usability scale. In Masaaki Kurosu, editor, *Proceedings of Human Centered Design*, HCI '09, pages 94–103, Berlin, Heidelberg, 2009. Springer Berlin / Heidelberg.

Thomas D. C. Little. Time-based media representation and delivery. In John F. Koegel, editor, *Multimedia Systems*, pages 175–200. ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 1994.

C. Karen Liu, Aaron Hertzmann, and Zoran Popović. Learning physics-based motion style with nonlinear inverse optimization. *ACM Transactions on Graphics*, 24(3):1071–1081, July 2005.

Noah Lockwood and Karan Singh. Biomechanically-inspired motion path editing. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '11, pages 267–276, New York, NY, USA, 2011. ACM.

Noah Lockwood and Karan Singh. Finger walking: Motion editing with Contact-Based hand performance. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '12, pages 43–52, Aire-la-Ville, Switzerland, 2012. Eurographics Association.

Jock D. Mackinlay, Stuart K. Card, and George G. Robertson. Rapid controlled movement through a virtual 3D workspace. *Computer Graphics*, 24(4):171–176, September 1990.

Chen Mao, Shengfeng Qin, and David K. Wright. A Sketch-Based gesture interface for rough 3D stick figure animation. In *Proceedings of Eurographics Workshop on Sketch Based Interfaces and Modeling*, 2005.

Paul Marino. *3D Game-Based Filmmaking: The Art of Machinima*. Paraglyph Press, Scottsdale, Arizona, 2004.

Anthony Martinet, Géry Casiez, and Laurent Grisoni. 3D positioning techniques for multi-touch displays. In *Proceedings of the 16th ACM Symposium on*

Virtual Reality Software and Technology, VRST '09, pages 227–228, New York, NY, USA, 2009. ACM.

Anthony Martinet, Gery Casiez, and Laurent Grisoni. The design and evaluation of 3D positioning techniques for multi-touch displays. In *Proceedings of IEEE Symposium on 3D User Interfaces*, 3DUI '10, pages 115–118, Los Alamitos, CA, USA, March 2010a. IEEE Computer Society.

Anthony Martinet, Géry Casiez, and Laurent Grisoni. The effect of DOF separation in 3D manipulation tasks with multi-touch displays. In *Proceedings of the 17th ACM Symposium on Virtual Reality Software and Technology*, VRST '10, pages 111–118, New York, NY, USA, 2010b. ACM.

Anthony Martinet, Géry Casiez, and Laurent Grisoni. Integrality and separability of multitouch interaction techniques in 3D manipulation tasks. *IEEE Transactions on Visualization and Computer Graphics*, 18(3):369–380, March 2012.

Justin Matejka, Tovi Grossman, Jessica Lo, and George Fitzmaurice. The design and evaluation of multi-finger mouse emulation techniques. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '09, pages 1073–1082, New York, NY, USA, 2009. ACM.

James McCann, Nancy S. Pollard, and Siddhartha Srinivasa. Physics-based motion retiming. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '06, pages 205–214, Aire-la-Ville, Switzerland, 2006. Eurographics Association.

Alberto Menache. *Understanding Motion Capture for Computer Animation*. Morgan Kaufmann, second edition, 2011.

William J. T. Mitchell. Spatial form in literature: Toward a general theory. *Critical Inquiry*, 6(3):539–567, 1980.

Tomer Moscovich and John F. Hughes. Multi-finger cursor techniques. In *Proceedings of Graphics Interface*, GI '06, pages 1–7, Toronto, Ontario, Canada, 2006. Canadian Information Processing Society.

Tomer Moscovich, Takeo Igarashi, Jun Rekimoto, Kentaro Fukuchi, and John F. Hughes. A multi-finger interface for performance animation of deformable drawings. In *Adjunct Proceedings of the Annual ACM Symposium on User Interface Software and Technology*, UIST '05, New York, NY, USA, October 2005. ACM.

Tomohiko Mukai and Shigeru Kuriyama. Pose-timeline for propagating motion edits. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '09, pages 113–122, New York, NY, USA, 2009. ACM.

Miguel A. Nacenta, Patrick Baudisch, Hrvoje Benko, and Andrew D. Wilson. Separability of spatial manipulations in multi-touch interfaces. In *Proceedings of Graphics Interface*, GI '09, pages 175–182, Toronto, Ontario, Canada, 2009. Canadian Information Processing Society.

Anja Naumann, Jörn Hurtienne, JohannHabakuk Israel, Carsten Mohs, Martin-Christof Kindsmüller, HerbertA Meyer, and Steffi Hußlein. Intuitive use of user interfaces: Defining a vague concept. In Don Harris, editor, *Engineering Psychology and Cognitive Ergonomics*, volume 4562 of *Lecture Notes in Computer Science*, pages 128–136. Springer Berlin Heidelberg, 2007.

Dennis C. Neale and John M. Carroll. The role of metaphors in user interface design. In Martin G. Helander, Thomas K. Landauer, and Prasad V. Prabhu, editors, *Handbook of Human-Computer Interaction*, pages 441–462. North-Holland, Elsevier, Amsterdam, second edition, 1997.

Michael Neff, Irene Albrecht, and Hans-Peter Seidel. Layered performance animation with correlation maps. *Computer Graphics Forum*, 26(3):675–684, 2007.

Nam Nguyen, Nkenge Wheatland, David Brown, Brian Parise, C. Karen Liu, and Victor Zordan. Performance capture with physical interaction. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '10, pages 189–195, Aire-la-Ville, Switzerland, 2010. Eurographics Association.

Laurence Nigay and Joëlle Coutaz. A design space for multimodal systems: Concurrent processing and data fusion. In *Proceedings of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems*, CHI '93, pages 172–178, New York, NY, USA, 1993. ACM.

Daisuke Ninomiya, Kohji Miyazaki, and Ryohei Nakatsu. Networked virtual marionette theater. In Ignac Lovrek, RobertJ Howlett, and LakhmiC Jain, editors, *Knowledge-Based Intelligent Information and Engineering Systems*, volume 5177 of *Lecture Notes in Computer Science*, pages 440–447. Springer Berlin Heidelberg, 2008.

Donald A. Norman. *The Design of Everyday Things*. Basic Books, New York, reprint paperback edition, 2002.

Chris North, Tim Dwyer, Bongshin Lee, Danyel Fisher, Petra Isenberg, George G. Robertson, and Kori Inkpen. Understanding multi-touch manipulation for surface computing. In *Proceedings of Human-Computer Interaction*, INTERACT '09, pages 236–249, Berlin, Heidelberg, 2009. Springer-Verlag.

Number None, Inc. *Braid*. Microsoft Game Studios, 2008.

Sageev Oore, Demetri Terzopoulos, and Geoffrey Hinton. A desktop input device and interface for interactive 3D character animation. In *Proceedings of Graphics interface*, GI '02, pages 133–140, Toronto, Ontario, Canada, 2002a. Canadian Information Processing Society.

Sageev Oore, Demetri Terzopoulos, and Geoffrey Hinton. Local physical models for interactive character animation. *Computer Graphics Forum*, 21:337–346, 2002b.

Masaki Oshita. Pen-to-mime: Pen-based interactive control of a human figure. *Computers & Graphics*, 29(6):931–945, December 2005.

Ken Perlin. Real time responsive animation with personality. *IEEE Transactions on Visualization and Computer Graphics*, 1(1):5–15, March 1995.

PIXAR. *25 Years of Animation*. Chronicle Books, 2010.

Suporn Pongnumkul, Jue Wang, Gonzalo Ramos, and Michael Cohen. Content-aware dynamic timeline for video browsing. In *Proceedings of the Annual ACM Symposium on User Interface Software and Technology*, UIST '10, pages 139–142, New York, NY, USA, 2010. ACM.

Jovan Popović, Steven M. Seitz, Michael Erdmann, Zoran Popović, and Andrew Witkin. Interactive manipulation of rigid body simulations. In *Proceedings of the Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '00, pages 209–217, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co.

Jovan Popović, Steven M. Seitz, and Michael Erdmann. Motion sketching for control of rigid-body simulations. *ACM Transactions on Graphics*, 22(4): 1034–1054, October 2003.

Jason L. Reisman, Philip L. Davidson, and Jefferson Y. Han. A screen-space formulation for 2D and 3D direct manipulation. In *Proceedings of the Annual ACM Symposium on User Interface Software and Technology*, UIST '09, pages 69–78, New York, NY, USA, 2009. ACM.

Jagannathan Sampath. Nuke. `http://sjagannathan.tripod.com/nuke/` (last accessed 6th January 2019), April 1999.

Ryan Schmidt, Karan Singh, and Ravin Balakrishnan. Sketching and composing widgets for 3D manipulation. *Computer Graphics Forum*, 27(2):301–310, 2008.

Johannes Schöning, Peter Brandl, Florian Daiber, Florian Echtler, Otmar Hilliges, Jonathan Hook, Markus Löchtefeld, Nima Motamedi, Laurence Muller, Patrick Olivier, Tim Roth, and Ulrich von Zadowv. Multi-Touch surfaces: A technical guide. Technical Report TUM-I0833, TU München, 2008.

Johannes Schöning, Frank Steinicke, Antonio Krüger, Klaus Hinrichs, and Dimitar Valkov. Bimanual interaction with interscopic Multi-Touch surfaces. In *Proceedings of Human-Computer Interaction*, INTERACT '09, pages 40–53, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.

Christian Schumacher, Bernhard Thomaszewski, Stelian Coros, Sebastian Martin, Robert Sumner, and Markus Gross. Efficient simulation of example-based materials. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '12, pages 1–8, Aire-la-Ville, Switzerland, 2012. Eurographics Association.

Stacey Scott, Sheelagh Carpendale, and Kori Inkpen. Territoriality in collaborative tabletop workspaces. In *Proceedings of the 2004 ACM conference on Computer supported cooperative work*, CSCW '04, pages 294–303, New York, NY, USA, 2004. ACM.

Andrew Sears and Ben Shneiderman. High precision touchscreens: design strategies and comparisons with a mouse. *International Journal of Man-Machine Studies*, 34(4):593–613, April 1991.

Rajvi Shah and P. J. Narayanan. Trajectory based video object manipulation. In *IEEE International Conference on Multimedia and Expo*, ICME '11, pages 1–4, 2011.

Hyun J. Shin, Jehee Lee, Sung Y. Shin, and Michael Gleicher. Computer puppetry: An importance-based approach. *ACM Transactions on Graphics*, 20 (2):67–94, April 2001.

Takaaki Shiratori and Jessica K. Hodgins. Accelerometer-based user interfaces for the control of a physically simulated character. *ACM Transactions on Graphics*, 27(5):123:1–123:9, December 2008.

Ben Shneiderman. The future of interactive systems and the emergence of direct manipulation. *Behaviour & Information Technology*, 1(3):237–256, July 1982.

Ben Shneiderman. Direct manipulation: A step beyond programming languages. *IEEE Computer*, 16(8):57–69, May 1983.

Ben Shneiderman. Direct manipulation for comprehensible, predictable and controllable user interfaces. In *Proceedings of the 2nd international conference on Intelligent user interfaces*, IUI '97, pages 33–39, New York, NY, USA, 1997. ACM.

Ken Shoemake. ARCBALL: a user interface for specifying three-dimensional orientation using a mouse. In *Proceedings of Graphics interface*, GI '92, pages 151–156, San Francisco, CA, USA, 1992. Morgan Kaufmann Publishers Inc.

Scott S. Snibbe. A direct manipulation interface for 3D computer animation. *Computer Graphics Forum*, 14(3):271–283, August 1995.

Ralf Steinmetz and Klara Nahrstedt. *Multimedia Fundamentals: Media Coding and Content Processing*, volume 1. Prentice-Hall, 2002.

David J. Sturman. *Whole-hand Input*. PhD thesis, Massachusetts Institute of Technology, 1991.

David J. Sturman. Computer puppetry. *Computer Graphics and Applications*, 18(1):38–45, February 1998.

Sandra Sudarsky and Donald House. An integrated approach towards the representation, manipulation and reuse of Pre-Recorded motion. In *Proceedings of Computer Animation*, CA '00, pages 56+, Los Alamitos, CA, USA, 2000. IEEE Computer Society.

Anders Svensson, Staffan Björk, and Karl P. Åkesson. Tangible handimation real-time animation with a sequencer-based tangible interface. In *Proceedings of the 5th Nordic Conference on Human-computer Interaction: Building Bridges*, NordiCHI '08, pages 547–550, New York, NY, USA, 2008. ACM.

Richard Taylor. *Encyclopedia of Animation Techniques*. Focal Press, 1996.

Sílvio C. Terra and Ronald A. Metoyer. Performance timing for keyframe animation. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '04, pages 253–258, Aire-la-Ville, Switzerland, 2004. Eurographics Association.

Sílvio C. Terra and Ronald A. Metoyer. A performance-based technique for timing keyframe animations. *Graph. Models*, 69:89–105, March 2007.

Lucia Terrenghi, David Kirk, Abigail Sellen, and Shahram Izadi. Affordances for manipulation of physical versus digital media on interactive surfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '07, pages 1157–1166, New York, NY, USA, 2007. ACM.

Frank Thomas and Ollie Johnston. *The Illusion of Life – Disney Animation*. Disney Editions, New York, NY, USA, 1981.

Matthew Thorne, David Burke, and Michiel van de Panne. Motion doodles: an interface for sketching character motion. *ACM Transactions on Graphics*, 23 (3):424–431, August 2004.

Adrien Treuille, Yongjoon Lee, and Zoran Popović. Near-optimal character animation with continuous control. *ACM Transactions on Graphics*, 26(3): 7:1–7:7, July 2007.

Philip Tuddenham, David Kirk, and Shahram Izadi. Graspables revisited: multi-touch vs. tangible input for tabletop displays in acquisition and manipulation tasks. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '10, pages 2223–2232, New York, NY, USA, 2010. ACM.

Edward R. Tufte. *Envisioning Information*. Graphics Press, Cheshire, Connecticut, 1990.

Ubisoft. *Prince of Persia: Sands of Time*. Ubisoft, Sony Computer Entertainment, 2003.

Brygg Ullmer and Hiroshi Ishii. Emerging frameworks for tangible user interfaces. *IBM Systems Journal*, 39(3.4):915–931, July 2000.

Andries van Dam. Post-WIMP user interfaces. *Communications of the ACM*, 40(2):63–67, February 1997.

Virgilio Vasconcelos. *Blender 2.5 Character Animation Cookbook*. Packt Publishing, 2011.

Anna Vögele, Max Hermann, Björn Krüger, and Reinhard Klein. Interactive steering of mesh animations. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '12, pages 53–58, Aire-la-Ville, Switzerland, 2012. Eurographics Association.

Benjamin Walther-Franks, Marc Herrlich, Markus Aust, and Rainer Malaka. Left and right hand distinction for Multi-Touch displays. In *Proceedings of the International Symposium on Smart Graphics*, pages 155–158, Berlin, Heidelberg, July 2011a. Springer-Verlag.

Benjamin Walther-Franks, Marc Herrlich, and Rainer Malaka. A Multi-Touch system for 3D modelling and animation. In *Proceedings of the International Symposium on Smart Graphics*, Berlin, Heidelberg, July 2011b. Springer-Verlag.

Benjamin Walther-Franks, Marc Herrlich, Thorsten Karrer, Moritz Wittenhagen, Roland Schröder-Kroll, Rainer Malaka, and Jan Borchers. DRAGIMATION: Direct manipulation keyframe timing for performance-based animation. In *Proceedings of Graphics Interface*, GI '12, Toronto, Ontario, Canada, 2012. Canadian Information Processing Society.

Jue Wang, Steven M. Drucker, Maneesh Agrawala, and Michael F. Cohen. The cartoon animation filter. *ACM Transactions on Graphics*, 25(3):1169–1173, July 2006.

Alan Watt and Fabio Policarpo. *3D Games: Animation and Advanced Real-time Rendering*. Addison-Wesley, 2003.

Daniel Wigdor and Dennis Wixon. *Brave NUI World: Designing Natural User Interfaces for Touch and Gesture*. Morgan Kaufmann, April 2011.

Richard Williams. *The Animator's Survival Kit*. faber and faber, expanded edition, 2009.

Andrew D. Wilson. Simulating grasping behavior on an imaging interactive surface. In *Proceedings of the International Conference on Interactive Tabletops and Surfaces*, ITS '09, New York, NY, USA, 2009. ACM.

Andrew D. Wilson, Shahram Izadi, Otmar Hilliges, Armando G. Mendoza, and David Kirk. Bringing physics to the surface. In *Proceedings of the Annual ACM Symposium on User Interface Software and Technology*, UIST '08, pages 67–76, New York, NY, USA, 2008. ACM.

Andrew Witkin and Zoran Popović. Motion warping. In *Proceedings of the Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '95, pages 105–108, New York, NY, USA, 1995. ACM.

Jacob O. Wobbrock, Meredith R. Morris, and Andrew D. Wilson. User-defined gestures for surface computing. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '09, pages 1083–1092, New York, NY, USA, 2009. ACM.

Mike Wu, Chia Shen, Kathy Ryall, Clifton Forlines, and Ravin Balakrishnan. Gesture registration, relaxation, and reuse for Multi-Point Direct-Touch surfaces. In *Proceedings of International Workshop on Horizontal Interactive Human Computer Systems*, TABLETOP '06, pages 185–192, Washington, DC, USA, 2006. IEEE Computer Society.

Yuan Wu, Hongxin Zhang, Chao Song, and Hujun Bao. Space-Time curve analogies for motion editing. In Falai Chen and Bert Jüttler, editors, *5th International Conference on Advances in Geometric Modeling and Processing*, pages 437–449. Springer Berlin / Heidelberg, 2008.

Katsu Yamane and Yoshihiko Nakamura. Natural motion animation through constraining and deconstraining at will. *IEEE Transactions on Visualization and Computer Graphics*, 9(3):352–360, 2003.

Ji S. Yi, Youn A. Kang, John T. Stasko, and Julie A. Jacko. Toward a deeper understanding of the role of interaction in information visualization. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1224–1231, November 2007.

KangKang Yin and Dinesh K. Pai. FootSee: an interactive animation system. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '03, pages 329–338, Aire-la-Ville, Switzerland, 2003. Eurographics Association.

Robert C. Zeleznik, Andrew S. Forsberg, and Paul S. Strauss. Two pointer input for 3D interaction. In *Proceedings of the Symposium on Interactive 3D Graphics*, I3D '97, New York, NY, USA, 1997. ACM.

David Zeltzer. Towards an integrated view of 3-D computer animation. *The Visual Computer*, 1(4):249–259, 1985.

Shumin Zhai. *Human Performance in Six Degree of Freedom Input Control*. PhD thesis, Department of Industrial Engineering, University of Toronto, Canada, 1995.

Shumin Zhai and Paul Milgram. Quantifying coordination in multiple DOF movement and its application to evaluating 6 DOF input devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '98, pages 320–327, New York, NY, USA, 1998. ACM Press/Addison-Wesley Publishing Co.

Peng Zhao and Michiel van de Panne. User interfaces for interactive control of physics-based 3D characters. In *I3D '05: Proceedings of the 2005 symposium on Interactive 3D graphics and games*, pages 87–94, New York, NY, USA, 2005. ACM.

Revisions

The original reviewed manuscript received two major and several minor revisions. The work of Karrer (2013) was referenced to reflect its relevance to this dissertation. This involved additions and changes in several places, most notably section 3.1. A digital appendix containing videos of the prototypes discussed in chapters 5 and 6 now provides additional illustration of the presented techniques. Reviewer names and viva date were included on the title page and the funding source was acknowledged. Finally, minor textual improvements, clarifications, and corrections were made throughout the text.

Digital Appendix

Contents

- Video demonstrating the multi-touch animation system (chapter 5)
- Video demonstrating the performance timing technique (chapter 6)