

Doctoral Thesis

**Motion and Vibration Effects Synthesis for
4D Films**

Jaebong Lee (이재봉)

Department of Computer Science and Engineering

Pohang University of Science and Technology

2015

**4D 영화를 위한
모션 및 진동 효과의 자동 생성**

**Motion and Vibration Effects Synthesis for
4D Films**

Motion and Vibration Effects Synthesis for 4D Films

by

Jaebong Lee

Department of Computer Science and Engineering

POHANG UNIVERSITY OF SCIENCE AND TECHNOLOGY

A thesis submitted to the faculty of Pohang University of Science
and Technology in partial fulfillment of the requirements for the
degree of Doctor of Philosophy in the Department of Computer
Science and Engineering

Pohang, Korea

September 21, 2015

Approved by


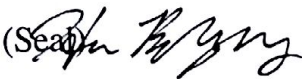




Seungmoon Choi, Academic Advisor

Motion and Vibration Effects Synthesis for 4D Films

Jaebong Lee

The undersigned have examined this dissertation and hereby certify
that it is worthy of acceptance for a doctoral degree from POSTECH.

9/21/2015

Committee Chair	최승문	(Seal)	
Member	한보형	(Seal)	
Member	이승용	(Seal)	
Member	홍기상	(Seal)	
Member	이성길	(Seal)	

DCSE 이 재 봉 Jaebong Lee, Motion and Vibration Effects Synthesis for
20120988 4D Films. 4D 영화를 위한 모션 및 진동 효과의 자동 생성,
Department of Computer Science and Engineering, 2015, 116P,
Advisor: Seungmoon Choi. Text in English

Abstract

4D film is an immersive entertainment system that presents various physical effects with a film. Despite the recent emergence of 4D theaters, production of 4D effects relies on manual authoring and remains laborious. To enhance the productivity of 4D effects production, we present algorithms that synthesize motion and vibration effects from the audiovisual content of a film.

We classified various motion and vibration effects into different categories and developed synthesis algorithms for each class of motion effects. The first class is for special effects, such as explosions and collisions, and our algorithm uses audio for the synthesis of impulses and vibrations. Unlike previous signal-level audio-to-vibration conversion methods, our algorithm considers only perceptual characteristics, such as loudness and roughness, of audio. This perception-level approach allows for designing intuitive and explicit conversion models with clear understandings of their perceptual consequences. The second class of effects is related to camera motion and it has two subclasses. The first subclass is those responding to fast camera motion to enhance the immersiveness of point-of-view shots, delivering fast and dynamic vestibular feedback. The second subclass moves viewers as closely as possible to the trajectory of slowly moving camera. Such motion provides an illusionary effect of observing the scene from a distance while moving slowly within the scene. For these two camera related subclasses, our algorithms compute the relative camera motion using computer vision techniques and then map it to a motion command to a 4D chair using appropriate motion mapping. The third class of motion effects tracks the motion of an object of interest. An object tracking algorithm estimates the position of the target

object, and then motion effects are synthesized according to the estimated position. The conversion between the position of the object and the motion commands to the 4D chair is performed by a viewer-centered rendering that matches the chair motion to the movement of the viewer's visual attention.

We assessed the subjective quality of our algorithms by user experiments, and all the results indicated that they are able to provide compelling 4D effects, sometimes even comparable to those manually designed by expert 4D designers.

Contents

1	Introduction	1
1.1	Paper Overview	2
1.2	Contributions	4
2	Related Work	6
2.1	4D Systems and Hardware	6
2.1.1	Motion Simulators and Control Algorithms	7
2.1.2	Vibration Displays and Rendering Methods	9
2.2	4D Effects Authoring Tools	11
2.3	Automatic Generation of 4D Effects	12
3	Surveys on 4D Effects	14
3.1	Classification of 4D Effects	14
3.2	4D Effects Production	18
3.3	Summary	20
4	Synthesis of Impulse and Vibration Effects	21
4.1	Translation System Overview	21
4.2	Auditory Loudness and Roughness	23
4.3	Vibrotactile Intensity and Roughness	25
4.3.1	Experimental Design	26
4.3.2	Results	27
4.3.3	Discussion	30

4.4	Perception Translation Model	32
4.5	Motion Effects Rendering for Impulse and Vibration	35
4.6	Accuracy of 4D Effect Triggering	36
4.7	User Experiment	37
4.7.1	Methods	37
4.7.2	Results	39
4.7.3	Discussion	40
4.8	General Discussion	42
5	Synthesis of Camera Class Motion Effects	44
5.1	Synthesis Algorithms	44
5.1.1	Camera Motion Estimation	44
5.1.2	Synthesis of Fast Camera Motion Effects	48
5.1.3	Synthesis of Slow Camera Motion Effects	52
5.1.4	Default Parameters and Implementation Issues	54
5.2	Results	55
5.2.1	Camera Motion Estimation and Motion Effects	55
5.2.2	Computational Complexity	59
5.3	User Experiments	60
5.3.1	Experiment I: General Users	60
5.3.2	Experiment II: 4D Experts	64
5.4	Discussion	66
6	Synthesis of Object Class Motion Effects	68
6.1	Overview of Interactive Motion Effects Design	68
6.2	Rendering algorithms for Object Motion	69
6.2.1	Object- and Viewer-Centered Rendering	69
6.2.2	Comparative User Study	71
6.3	Implementation of Viewer-Centered Rendering	76
6.3.1	Washout Filter Algorithm	76
6.3.2	Selection of Object Tracking Algorithm	77
6.3.3	Motion Effects Design Using Object Tracking	79
6.3.4	Motion Effects Design Using Spline	83
6.4	User Experiments	87
6.4.1	Experiment I: General Users	87

6.4.2 Experiment II: 4D Experts	89
7 Conclusions	92
한글 요약문	94
REFERENCES	96

List of Figures

1.1	Examples of 4D related products.	2
1.2	Audience watching a 4D film with motion effects synthesized by our algorithms.	3
2.1	Classical washout filter algorithm. This figure is reproduced from [60]. For simplicity, some parts related to transformation between reference frames are removed from the original figure.	8
2.2	Example of widely used motion platforms.	9
3.1	Frequencies of 4D effects appearing in regular 4D films.	15
3.2	Examples of six motion effect classes. (a) When Electro releases a strong electric field, a motion chair invokes a strong, rough vibration. (b) Motion effects are generated following fast and abrupt camera motion aligned with the virtual passenger riding a roller coaster. (c) A motion chair follows slow camera motion to provide an illusion that the viewer is looking at the earth in a spacecraft. (d) When Kristoff rides on a reindeer, a motion chair follows the continuous movement of Kristoff's body. (e) When Legolas strikes an enemy with a sword, a motion chair produces short, discrete feedback tracking the movement of the sword. (f) When Toretto hits the accelerator of a car, a motion chair tilts backward to provide the sensation of sudden acceleration.	16
3.3	Frequencies of the six classes of motion effects.	18

4.1	Overview of our perception-level audio-to-tactile translation.	22
4.2	Loudness values computed for 20 different 4096-long audio samples (sampling rate 44.1 kHz) collected from various music and movie clips. The Glasberg's model and our simplified model (after constant scaling) produced similar results.	24
4.3	Posture of participants used in all experiments.	26
4.4	The perceived intensities and roughnesses measured from the psychophysical experiments. Amplitude 1 and Amplitude 2 represent the amplitudes of the 175 Hz and 210 Hz components, respectively.	28
4.5	Fitted perceived intensity and roughness functions. Dots represent the measured data.	29
4.6	Perceived intensities and roughnesses measured using a tablet.	31
4.7	Relationship between the auditory perceptual variables and the special sound effects accompanied in the game, Battlefield 1943 (EA Digital Illusions CE; Sweden). Loud background music is played continuously, so the loudness L_a shows small variance. However, the roughness R_a shows the peaks (highlighted by dotted ellipses) highly correlated to the explosions and gun fires labeled by (1)–(3).	32
4.8	Relationship between the auditory perceptual variables and the special sound effects accompanied in the movie, Cars (Pixar; USA). Loud engine sound is played during (1), and background music is played during (2). Even though the loudness metric cannot distinguish these two sounds, the roughness metric successfully detects only the strong engine sound.	33
4.9	Example results of our perception-based audio-to-vibrotactile translation algorithm.	34
4.10	Motion effects generated by Algorithm V (Wall of China; Simuline Inc.).	35
4.11	Results of the user experiment. The error bars represent standard errors.	40
4.12	Results of games after grouping the participants in regard to their gaming preference. The error bars represent standard errors.	41

5.1 Overview of our camera motion estimation algorithm and an example output of each step. Images (a) to (f) are example outputs taken from Amazon (ride film; Simuline Inc.). (a) to (d) are results obtained from the 607th and 608th frames. In (b), hue and brightness represent the direction and length of optical flow, respectively. In (c) and (d), flow vectors are represented by lines. In (d), blue lines represent inliers and red lines denote outliers. (e) and (f) are angular velocities estimated from the 510th to the 692th frames. Blue, red, and green lines represent angular velocities in roll, pitch, and yaw, respectively. 45

5.2 (a) Definition of 6-DOF motions. (b) Linear-to-angular approximation (surge to pitch). For small θ , $x \approx l = R\theta$, so $x \propto \theta$ 49

5.3 Motion synthesis algorithm for CF effects. 50

5.4 Motion synthesis algorithm for CS effects. 53

5.5 A comparison between measured and estimated camera motions (a, b) and synthesized CF motion effects (c). Highlighted parts (1)–(4) are also shown in upper images. Note that the high-frequency noise in the measured camera motion is injected from ambient noise in the car. 56

5.6 Comparison of surge motions estimated by our camera motion estimation algorithm and Boujou 5 before smoothing or filtering. The same video used in Fig. 5.5 was used as input. 57

5.7 Motion effects generated with Frozen by Algorithm CS with the equi-time segmentation. Red, blue, and green lines are for the first, second, and third segments, respectively. In (b), dashed lines are interpolated motions to connect the segments. A dotted ellipse represents the workspace of our motion chair. 58

5.8 Results of Experiment I with general users. Error bars represent standard errors. Note that the scale for fatigue is inverted so that the higher the fatigue score, the better the performance (less tiring), to be consistent with the others. The sets of motions effects marked with the same alphabets indicate that they did not show statistically significant differences by Tukey’s HSD test. 62

5.9 Examples of randomly-generated (RE), automatic (AE), and manually-designed (ME) motion effects (Fairy Balloon Ride; a ride film from Simuline Inc.). Blue and red boxes represent the dominant camera motions that were manually annotated by a human viewer. The blue boxes indicate horizontal motions (L: left turn and R: right turn; related to roll), and the red boxes are for vertical motions (D: descent and A: ascent; related to pitch). 63

5.10 Results of the user study with 4D experts. Error bars represent standard errors. The scale for fatigue is inverted as in Fig. 5.8. Asterisks indicate that the results between DG and OG had statistically significant differences. 65

6.1 Example user interface of interactive motion effects design. When the center of the object being tracked moves outside the yellow circle, the designer fixes its bounding box before clicking the ‘Track next frame’ button. 69

6.2 Object- and viewer-centered rendering in an example plane scene. 70

6.3 Results of comparative user study between object- and viewer-centered rendering. Among total 18 videos, only 8 statistically significant ($\alpha = 0.05$) results are illustrated. Results of remaining 10 videos are presented in Fig. 6.4. Error bars represent standard errors. Asterisks indicate that the two rendering methods were statistically significant. 74

6.4 Results of comparative user study between object- and viewer-centered rendering. Among total 18 videos, 10 statistically not significant results are illustrated. Error bars represent standard errors. 75

6.5 Washout filter algorithm for viewer-centered rendering. 77

6.6 The ratio of annotated frames with respect to various tracking error thresholds. The smaller is the better. MEEM was better than the other compared algorithms when evaluated using center location errors. (a) and (b) are the same results, but (b) shows the results in more detail for error thresholds 20–100 pixel. 78

6.7 Results of the user experiment for the variations of tracking error thresholds (in pixel). Error bars represent standard errors. The set of motion effects marked with the same alphabets indicates that they did not show statistically significant differences by the SNK tests. The table shows an average frame annotation ratio for each experimental condition. 80

6.8	The results of individual videos for different tracking error thresholds. Error bars represent standard errors. The set of motion effects marked with the same alphabets indicate that they did not show statistically significant difference by SNK test.	81
6.9	Examples of the motion effects generated by SCM and MEEM with different tracking error thresholds for <i>How to Train Your Dragon 2</i> . (a) shows motion effects synthesized based on the ground truth positions of the object.	82
6.10	Correlation coefficient between the motion effects generated by the ground truth and the motion effects generated by tracking with different error thresholds. Average correlation coefficient for five video clips were taken.	83
6.11	Results of the user experiment that compared the motion effects designed by object tracking and natural cubic spline. Error bars denote standard errors. The set of motion effects marked with the same alphabets represent that they did not show statistically significant differences by the SNK tests. The table shows average frame annotation ratios for each experimental condition.	84
6.12	The results of individual videos (The first three videos). Error bars denote standard errors. The set of motion effects marked with the same alphabets represent that they did not show statistically significant difference by SNK test.	85
6.13	The results of individual videos (The remaining two videos). The set of motion effects marked with the same alphabets represent that they did not show statistically significant difference by SNK test.	86
6.14	Results of Experiment I with general users. Error bars represent standard errors. Alphabets represent SNK grouping.	88
6.15	Results of the user study with 4D experts. The scores of harmony, comfort, and level for three video clips were averaged. D1, D2, D3, and D4 are the results of designers and E1 and E2 are the results of developers.	90

List of Tables

3.1	Ten regular 4D films used in the survey.	15
5.1	Parameter values for FlowLib v3.0.	54
5.2	Default b and c for Algorithm CF and CS.	54
5.3	Execution times (ms) for one frame.	59
6.1	List of videos used in the comparative user study. If object- and viewer-centered rendering were significantly different ($\alpha = 0.05$), the serial number of the video is marked with a bold text. Blue and red number represent object- and viewer-centered rendering was better, respectively.	72

Chapter 1

Introduction

4D film refers to an immersive entertainment system that presents various physical effects, such as motion, vibration, wind, water, and scent, with a 2D or 3D film. Evolving from popular attractions in amusement parks, 4D films are now produced from regular films and screened in 4D theaters. D-BOX, a major Canadian manufacturer of 4D motion chairs, has 113 theaters with its motion systems in the United States alone [14]. Another manufacturer based in Korea, CJ 4DPLEX, has opened 156 4D theaters in 33 countries and releases more than 50 4D films in a year [1, 13].

4D has been adopted not only for theaters but also for personal handheld devices and products for home use. For these products, vibration effects are commonly used alone without other physical effects because tactile displays are relatively simple and inexpensive. Game controllers, smartphones and tablets, portable media players, vibration earphones and headphones, and 4D home theaters are typical examples. Fig. 1.1 shows examples of 4D theater and 4D related products.

Recent growth of 4D industry has elevated the needs for efficient production methods of 4D effects. However, 4D effects designers still manually create all effects using in-house authoring software, and 4D effects production remains highly labor-intensive. For example, three 4D effects designers need to work for about 16 days to make a 4D film out of one regular film (see Section 3.2). In this situation, automated algorithms that synthesize



(a)4DX: 4D theater



(b)D-Box: motion platform for home theater



(c)RCraft: motion simulator for racing game



(d)Mophie Pulse: iPod accessory for tactile feedback



(e)Dual Shock: gaming controller with tactile feedback



(f)Crowson Shadow-8: tactile chair for home theater

Fig. 1.1: Examples of 4D related products.

4D effects from regular films by analyzing their video and audio streams, at least for partial clips, can improve productivity to the great extent. Such algorithms are also expected to allow 4D effects designers to spend effort on more artistic and creative tasks, thereby leading to better 4D effects. Further, automatic algorithms may be able to pioneer new 4D applications, e.g., live broadcasting of F1 racing or World Cup with real-time motion and vibration effects.

1.1 Paper Overview

The aim of our research is to contribute to 4D film production by providing autonomous synthesis algorithms of 4D effects. Among others, this work concentrates on motion and vibration effects, which are the most frequently used in 4D films. This specific goal was identified through surveys on 4D effects and their current production system (Chapter 3). The surveys enabled us to classify motion and vibration effects into six classes and summarize common design strategies for each class. Our motion effects synthesis algorithms were built upon the survey results.



Fig. 1.2 Audience watching a 4D film with motion effects synthesized by our algorithms.

Our algorithms assume that 4D effects designers segment a film into a number of audio-visual streams and determine the 4D effects that will be used for each segment (a procedure called scene breakdown; Section 3.2). Our algorithms process the audiovisual content of each segment to synthesize motion or vibration effects of the designated type. Five classes of effects that cover most of the 4D effects used in 4D films are currently supported.

The first class is the vibration and motion effects for special effects in a film, such as explosions and collisions. They are synthesized by our perception-level audio-to-vibrotactile translation algorithm (Chapter 4). This algorithm maps the perceptual variables extracted from an audio signal to the desired perceptual variables of vibrotactile stimuli. Since the perceptual factors, such as loudness, roughness, and brightness, have explicit perceptual and emotional meanings, this higher-layer framework is much more intuitive and easier for designers and users to understand the consequences of stimulus translation.

Next two classes of motion effects are generated from camera motion that is estimated from visual scenes using computer vision techniques (Chapter 5). The estimated camera motion is used to synthesize motion effects that respond to fast and abrupt camera motion (Section 5.1.2). These motion effects improve the immersiveness of point-of-view (POV) shots by delivering dynamic vestibular feedback, like 4D rides in amusement parks. When the camera moves very slowly in a long shot, our motion synthesis algorithm converts it to gentle and continuous movements of a motion chair (Section 5.1.3). Such effects make viewers feel as if they were looking at the scene in the viewpoint of a slowly moving camera, thereby improving presence—the feeling of being there. Both algorithms are designed for the limited workspace of motion chairs.

For the remaining two classes, motion effects track the motion of the object of interest (Chapter 6). An object tracking algorithm estimates the position of the target object and then motion effects are synthesized according to the estimated positions. The conversion between the object's position and motion command of a chair is performed by a viewer-centered rendering that tries to match the motion of the chair to the movement of visual attention of a viewer. Even though many annotations are required, motion effects design using our algorithm can be done at least 10 times faster than the current manual authoring.

In 4D films, different classes of motion effects are frequently produced at the same time. For example, when a flying plane is shot by an enemy fire, sudden movement of a chair is rendered with continuous motion of the chair that tracks motion of the plane. Therefore, motion effects synthesized by different algorithms should be added together to make the final motion effects.

1.2 Contributions

Our contributions are:

1. The comprehensive survey and analysis on 4D effects and its production process.
2. The three algorithms that synthesize motion and vibration effects in a similar way to the practice of 4D effects designers.

3. The thorough performance evaluations including user experiments.

In Chapter 4, we also proposed:

4. general perception-based audio-to-tactile translation framework, and
5. synthesis framework of superimposed vibrotactile stimuli with desired perceptual properties.

These results can also be applied to other HCI related platforms and applications.

Related Work

2.1 4D Systems and Hardware

The earliest example of 4D system is the Sensorama, which was developed in the late 1950s [25]. This system can play 3D motion picture with stereo sound, smell, vibration, and wind. Ever since then, 4D systems have become widespread as attractions in an amusement park and a museum [88, 86]. Audiovisual contents are also produced especially for this type of 4D systems. 4D systems were first applied to a regular film and screened in movie theaters by D-Box and CJ 4DPLEX in 2009. In the D-Box theaters, part of seats are designated as the D-Box reserved and only motion and vibration effects are provided. Whereas in the 4DX theaters (CJ 4DPLEX), the whole seats are motion chairs and additional environmental effects, such as water, wind, and strobe, are provided with motion and vibration effects.

Even though 4D systems were commercially successful, rigorous academic research on such 4D systems has not been very active, with only a few exceptions. For instance, Hirota et al. [26] presented a multi-sensory theater with olfactory, wind, and pneumatic displays, along with a content editing tool that used MIDI interfaces. Oh et al. [63] analysed how 4D effects affect presence of audience in terms of Lombard and Ditton's six conceptualizations of presence [54]. There has also been growing interest in realistic broadcasting system by adding various sensory modalities to conventional audiovisual contents [64, 91, 38].

Some of them constructed a broadcasting system with multisensory feedback based on the recent MPEG-V standard in which additional sensory effects and virtual objects can be stored [91, 38].

In virtual reality, there have been many studies on individual display technologies that can be integrated into 4D systems, such as vestibular (motion) [60, 83, 62], vibrotactile [50, 33], olfactory [61, 58], and wind displays [59, 57] (see [6] for a more general review on this topic). Among various 4D effects, motion and vibration effects are generally regarded as the most effective in improving immersiveness, so they are used most frequently (see Section 3.1). Our algorithms also cover motion and vibration effects only.

2.1.1 Motion Simulators and Control Algorithms

Motion simulators and their control algorithms were first developed as early as in 1970s for a flight simulation [76, 65] (Fig. 2.2a). These simulators were usually built based on the 6-DOF Stewart platform. Many motion control algorithms have been proposed, including the classical washout filter [76], the coordinated adaptive algorithm [65], and the optimal control algorithm [80]. However, a general consensus is that the classical washout filter provides the best trade-off between perceptual quality and algorithmic simplicity [60].

Fig. 2.1 shows the classical washout filter algorithm for 6-DOF motion simulators. The inputs are body-axis specific forces, f , and angular rates, w . Specific forces are defined as $f = a - g$, where a is linear accelerations and g is a gravitational acceleration. The outputs are positions, S , and Euler angles, β of the simulator.

The classical washout filter is essentially composed of a set of high-pass filters and low-pass filters. High-pass filters remove low frequency motions that tend to move motion platform to its workspace limits, while maintaining high frequency motions unchanged. The high-pass filters also serve to bring the motion platform back to its initial position to obtain simulation space for next motions; this is the reason for the name of the algorithm, ‘washout’ filter. Third-order translational and second-order rotational high-pass filters are sufficient to fulfill ‘washout’ even for steady translational acceleration or rotational velocity. In practice however, lower-order filters, second-order translational and first-order rotational

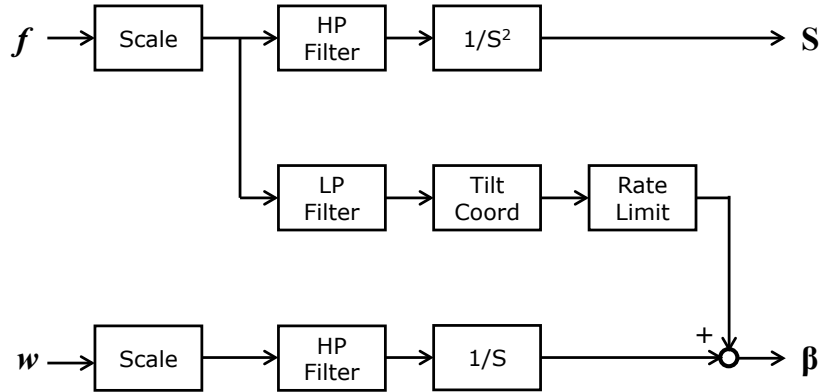
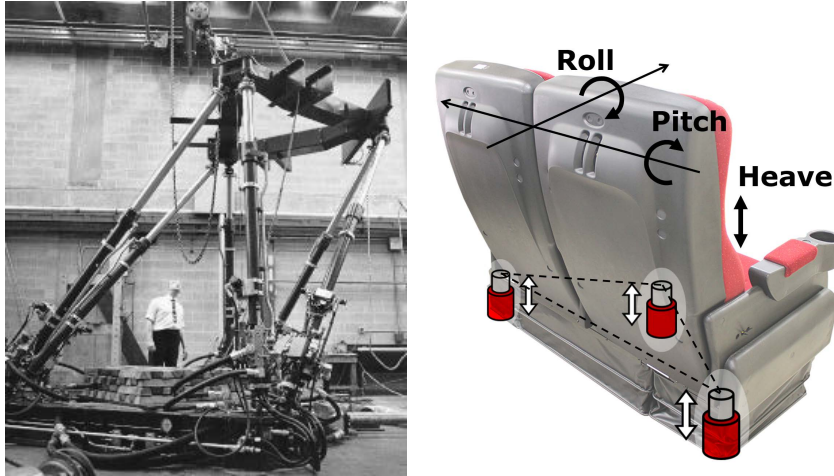


Fig. 2.1 Classical washout filter algorithm. This figure is reproduced from [60]. For simplicity, some parts related to transformation between reference frames are removed from the original figure.

filters, are usually sufficient and more preferred since these severe cases are very rare. The linear accelerations are also passed to low-pass filters to render sustained accelerations by using a gravity vector trick, which substitutes gravity for linear acceleration. For example, when a constant forward acceleration needs to be rendered, the motion platform tilts backward to use the gravity vector as a replacement for the correct inertial force induced by forward acceleration. Generally, this gravitational force cannot be perceived differently from being constantly accelerated without additional sensory information. However, changing the tilt angle can be perceived as undesirable angular motion, so the rate limit (Fig. 2.1) suppresses the angular rate below the threshold (3 deg/s is a widely accepted for this purpose).

Motion simulators began to be adopted for entertainment from the mid-1980s [88]. The classical washout filter is a de facto standard algorithm for motion platforms for entertainment. Even though various motion platforms with different configurations have been used, a 3-DOF platform with roll, pitch, and heave motion is most widely used for entertainment [86], because it can be manufactured more cheaply than a 6-DOF platform by using only three linear actuators (Fig. 2.2b). Moreover, this 3-DOF motion platform has proven capability that can produce motion effects with comparable quality to that generated by a 6-DOF Stewart platform [67].



(a)The first flight simulator based on a 6-DOF Stewart platform (b)Typical 3-DOF motion platform for 4D film

Fig. 2.2 Example of widely used motion platforms.

Another noteworthy attempts on motion platform have been made by Danieau and colleagues. They designed a low-cost motion chair, which stimulates the viewer's two hands and head using three force-feedback devices, and evaluated the quality of experience (QoE) of that approach [16]. They also introduced a concept of haptic cinematography and an associated taxonomy [17]. As a proof of concept, they designed haptic rendering models for several camera effects (e.g., zoom-in and tilting) that are frequently used in cinematography based on the assumption that camera motion is known, along with an assessment of their QoE.

2.1.2 Vibration Displays and Rendering Methods

Vibration is the second most frequently used effects in 4D films (see Section 3.1). In 4D theater, a motion chair is usually equipped with vibration actuators to generate high frequency vibration that cannot be generated by lower bandwidth of motion actuators.

Many research groups developed novel systems with vibrotactile actuators to deliver immersive multimedia experience. Kim et al. designed a tactile glove with an array of vibration motors to provide tactile sensations synchronized with a multimedia content [42].

Their tactile movie system stores spatio-temporal tactile data as a gray scale video in the MPEG-4 framework. Israr and Poupyrev developed the Tactile Brush algorithm that can generate two dimensional continuous moving sensation on the surface of the skin [33]. It can create smooth moving sensation with varying frequency, intensity, velocity, and direction of motion using sparse grid of vibration actuators by utilizing two well-known tactile illusions, apparent haptic motion [78] and phantom sensation [3]. Kim et al. [39] also used a chair with a tactile grid to enhance audiovisual experience. Lemmens et al. [50] and Rahman et al. [68] developed a wearable tactile jacket with many vibration motors for the same purpose. Interestingly, Lemmens et al. designed vibrotactile patterns based on a common sayings, for example, vibrations along the spine used to deliver fear owing to ‘a shiver down one’s spine’. Lee et al. [47] attached tactile array on the forearm in order to provide location of a ball in a soccer game. Most of these works performed user experiments and the results suggest that tactile effects help the user to be more immersed in the audiovisual content.

Vibration is also widely adopted in mobile devices, due to its low cost, small size, simplicity, and high effectiveness. Kim et al. [40] and Seo and Choi [77] present vibrotactile rendering methods that make a moving sensation along the surface of a mobile device based on the tactile illusions. The former controls the actuation time difference and the latter modulates amplitude of each actuator to generate moving vibration. They assessed the effectiveness of their approach by implementing a ball rolling game and a bricks-breaking game. Hwang et al. [32] adopted dual-mode actuator, which can produce vibrations with two principal frequencies, to enhance music listening experience. In industry, Immersion Corp. has TouchSense technology that offers integrated solution for tactile content production on mobile devices. This technology has been widely adopted in smart phones and gaming controllers.

There are two review papers that present recent and comprehensive review about this topic. [12] provides a general introduction about vibrotactile display and [18] focuses on haptic technology used to enhance audiovisual experience.

2.2 4D Effects Authoring Tools

To manually design 4D effects, specialized authoring tools are required. Several research groups have developed authoring tools for vibrotactile pattern [74, 49, 42, 28, 34], 4D theater [26], and 4D broadcasting [41, 84]. [74, 26, 41, 84] share similar interfaces providing multiple timelines, each of which is for an individual 4D effect that 4D designers need to create.

Several research outcomes made an attempt to adopt novel interface to design vibrotactile effects. The VibScoreEditor [49] employs a metaphor of musical score for symbolic design, motivated by the significant similarity between sound and vibration. It adopts most of musical notations as it is, for example, higher position of a note in the staff lines denotes a higher frequency (in the musical score, it means higher tone). However, the intensity of vibration is denoted by a number in head of a note instead of standard musical notations for strength, such as *pianissimo* (very soft) and *forte* (loud). It is because these terms are subjective and they cannot be used for every single note. The authoring tool of Kim et al. [40] allows a user can create multichannel, spatially distributed tactile stimuli by drawing lines on video display window with a tactile brush. The user also can design a temporal tactile trajectory easily by drawing lines while the video is being played slowly. The demonstration-based authoring method [28] integrates various touch input properties, such as the position and pressure of a touch, to generate complex vibrotactile pattern. The conversion between touch and vibration is performed in vary intuitive way. The duration of a press is mapped to the on-time of a vibration, the pressure is converted to the strength, and the vertical position of touch is mapped to the frequency of vibration. Zhao et al. developed FeelCraft Design Editor [93] based on a feel effects library [34] that allows explicit pairing between a meaningful linguistic phrase and a vibrotactile pattern. For example, a user can design vibrotactile pattern for rain intuitively by adjusting parameters such as amount of drops, size of drops, and force of drops instead of directly manipulating properties of vibrotactile pattern.

In industry, the Haptic Studio (Immersion; USA) is a standard program for vibrotactile

pattern authoring, especially for mobile devices. However, there is no standard tool for general 4D effects production. 4D film companies, e.g., D-Box Technologies (Canada) and CJ 4DPlex (Korea), use their in-house authoring programs to design 4D effects. These tools are also based on a common interface with multiple timelines. Even though they include some convenient functions, current 4D effects authoring is a fully manual task. The manual design process is time-consuming by nature, although it can be facilitated to some extent by useful user interface components such as a library of reusable 4D patterns.

2.3 Automatic Generation of 4D Effects

Automatic generation of 4D effects is indispensable for the further growth of 4D films and associated applications. Research in this direction is still at infancy, but it has seen increasing endeavors. For example, Chi et al. developed a sound-specific vibration interface that responds to only pre-learned target sounds and applied their interface to the gunfire effects of a commercial game [11]. The matching success rate was about 80%. Kim et al. developed a tactile chair system that emphasizes visually salient regions on the screen by vibrotactile feedback that is spatially mapped onto the viewer's back [39]. The visually salient areas are identified by a real-time GPU-accelerated algorithm. Hwang et al. introduce a dual-band haptic music player that convert audio signal to vibrotactile music in real-time [32]. They developed specialized algorithm for the dual-mode actuator and compared the perceptual quality of vibrotactile effects to effects generated by conventional LRA (Linear Resonant Actuator). They extended their work using the auditory saliency estimation to emphasize only the salient part in music by vibrotactile effects [31]. Further, Lee and Han proposed an early algorithm that determines the posture of motion platform based on block matching between two video frames [48].

A notable progress has been made very recently by Shin et al. [79], who presented a framework that shared the same motivation with our present work. Their framework relies on Boujou (a commercial matchmover; Vicon Motion Systems) to estimate the 3D trajectory of camera in the world coordinate frame from sequential 2D images. The camera position is then numerically differentiated once and twice to obtain angular velocity and

linear acceleration, respectively. These variables are fed to classical washout filters to make motion commands of a motion chair. To suppress the noise amplified by the numerical differentiation, they proposed a total variation-based noise reduction technique, which is a non-causal filter that finds an optimal balance between fitting error and jerk (the derivative of acceleration). They also project the direction of gravity progressively to the 3D scene for gravity rendering. The initial gravity direction can be estimated from the first image if the image contains prominent vertical lines, or it needs to be set by the user. They presented many motion effects synthesized by their framework, but reported no user studies that would allow for the objective assessment of perceptual quality.

Audio-to-vibrotactile conversion methods also have been developed actively for other applications, such as assisting hearing-impaired or human-computer interaction. Chang and O'Sullivan made audio-based haptic UI feedback for mobile devices using MFT (Multi-Function Transducer) as an actuator [10]. The vibrotactile feedback was enabled by extracting low frequency components from audio signals. Li et al. designed PeopleTone, which notifies buddies proximity using mobile phones via audio and/or vibrotactile cues [52]. The vibrotactile cues were generated from audio signals by taking running sum of band-pass filtered signal and exaggerating differences in signal strength. Karam et al. introduced Model Human Cochlea (MHC) that translates music into vibration signals to give musical expressions to the visually impaired [37]. They used an array of voice coil actuators attached to the back of a chair, and each actuator was assigned to a individual track of multi-track audio or a predefined frequency band. Audio-based autonomous vibrotactile feedback also been studied to add realistic tactile feedback to digital musical instruments [9, 7, 56]. Although these methods were not developed for 4D, they can be easily modified to generate vibration effects for 4D films.

Many products with audio-to-vibrotactile conversion technology are already available on the market, such as vibration earphones and headphones, ViviTouch (Artificial Muscle; USA), and Reverb module (Immersion; embedded in recent Samsung Galaxy S series), even though most of these products rely on a simple signal-level conversion method using a low-pass filter or a filter bank.

Chapter 3

Surveys on 4D Effects

Detailed guidelines for 4D effects designs are difficult to establish. After all, it is a creative artistic process that greatly depends on the expertise, experience, and preference of human designers. Under this circumstance, the first and foremost step of our research had to be defining the goals and requirements of algorithms that would afford the best benefits. To this end, we carried out two surveys, and results are described in this section.

3.1 Classification of 4D Effects

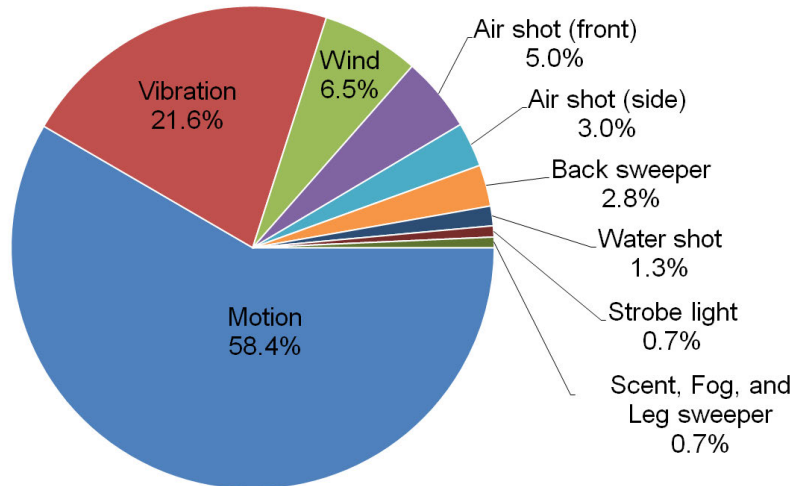
Our first survey was to classify the types and design methods of 4D effects provided in 4D films and observe their frequencies of use. At present, two kinds of 4D films are popular. One is *4D rides*, in which most scenes consist of POV shots for the best 4D experience. 4D effects are provided intensively, but in a short running time (less than ten minutes) to prevent viewers' fatigue. The other type is regular films to which 4D effects are added after production. Such *regular 4D films* are much longer and use more sparse 4D effects.

For this survey, we watched ten regular 4D films at 4DX theaters (CJ 4DPLEX) and eight 4D rides at various places, and then analyzed the 4D effects displayed with the films. Regular 4D films are generally made for action and adventure genres, and our selections shown in Table 3.1 are an adequate representative of regular 4D films.

The ten regular 4D films provided a total of 2278 4D effects. We counted the number of

Table 3.1 Ten regular 4D films used in the survey.

Title	Release Year
Percy Jackson: Sea of Monsters	2013
Gravity	2013
Thor: The Dark World	2013
Ender's Game	2013
Frozen	2013
The Hobbit: The Desolation of Smaug	2013
Need for Speed	2014
Captain America: The Winter Soldier	2014
The Amazing Spider-Man 2	2014
X-Men: Days of Future Past	2014

**Fig. 3.1** Frequencies of 4D effects appearing in regular 4D films.

times with which each 4D effect was presented, and their frequencies of use are visualized in Fig. 3.1. Motion effects were most frequently used (58.4%), while vibration effects ranked at the second (21.6%). The use of other effects was sporadic. These results indicate that motion effects deserve the highest priority for automatic synthesis, followed by vibration effects.

We further classify motion effects into six classes according to the grounds of motion effects. Typical examples of the six classes are also provided in Fig. 3.2.

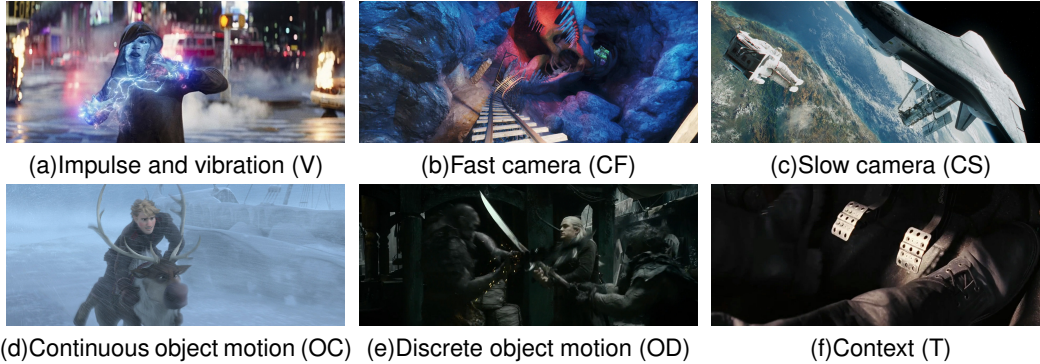


Fig. 3.2: Examples of six motion effect classes. (a) When Electro releases a strong electric field, a motion chair invokes a strong, rough vibration. (b) Motion effects are generated following fast and abrupt camera motion aligned with the virtual passenger riding a roller coaster. (c) A motion chair follows slow camera motion to provide an illusion that the viewer is looking at the earth in a spacecraft. (d) When Kristoff rides on a reindeer, a motion chair follows the continuous movement of Kristoff’s body. (e) When Legolas strikes an enemy with a sword, a motion chair produces short, discrete feedback tracking the movement of the sword. (f) When Toretto hits the accelerator of a car, a motion chair tilts backward to provide the sensation of sudden acceleration.

Impulse and vibration class (V): Motion effects in this class offer responses to impacts or vibrations in the scene, such as gun fire, explosion, or vehicle vibration. V effects are very short in time but they occasionally last for a few seconds, e.g., when expressing ambient vibrations inside a vehicle.

Camera motion class (C): Motion effects are generated by following the camera motion. It has two subclasses: *fast camera motion class (CF)* and *slow camera motion class (CS)*. CF effects are fast and abrupt, and deliver dynamic motion effects for POV shots. They are frequently used in 4D rides and films with racing scenes. CS effects present gently and continuously moving sensations when a slowly moving camera shoots landscape view from a distance. Viewers perceive an illusional effect that they observe the scene from a distance while moving together with the camera. Effect durations are generally long in this class (up to 30 s for class CS).

Object motion class (O): Motion effects track the motion of a character or an object of

interest, also with two subclasses: *continuous object motion class* (OC) and *discrete object motion class* (OD). OC effects target an object that moves continuously for a relatively long period of time, e.g., in running, chasing, driving, and flying scenes. OD effects represent short, discrete motions and are often used in fighting scenes. OD effects are very short (less than 1 s), while OC effects are usually longer (2–5 s). There can be an ambiguity between C and O effects when the camera and objects move simultaneously. In such cases, 4D effects designers determine which class of motion effects to use based on their subjective judgment of which motion is more dominant, also with contextual consideration.

OD effects can also be related to impacts like V effects. The key difference between OD and V effects is that OD effects produce clear directional cues while V effects are usually omnidirectional.

Context class (T): Certain motion effects are created based on the designers' understanding of the context of events. Current images or sounds do not provide direct clues. An example is when a scene shows only a driver turning the steering wheel to left, motion feedback is also exerted to left to simulate the expected movement of the car. T effects are relatively short, similar to O effects.

Fig. 3.3 shows the relative frequencies with which the six classes of motion effects appeared in the 18 4D films. The 4D rides relied on only CF and V motion effects. All classes of motion effects were observed in the regular 4D films. It is noted that even though O effects were observed more frequently than C effects, the latter has longer durations as described earlier. The actual playing time of C effects is comparable to or even exceeds that of O effects.

Vibration effects mostly correspond to the impulse and vibration (V) class. To render impacts or vibrations, vibration effects are used more frequently than motion effects. Motion effects are presented only for strong and long impacts or vibrations in the scene.

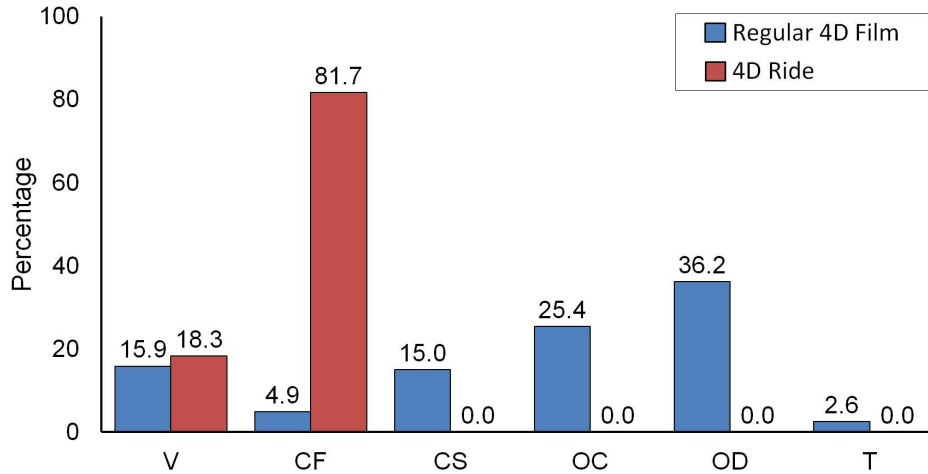


Fig. 3.3 Frequencies of the six classes of motion effects.

3.2 4D Effects Production

The second survey was an expert interview with three experienced 4D effects designers who worked for CJ 4DPLEX, in order to learn the general production procedure of 4D films and common design heuristics. They had two to four years of experience, and have produced more than 100 4D films.

According to the interviewees, 4D effects production is divided into three stages: scene breakdown, editing, and revision. In scene breakdown (also called pre-production), designers make an overall plan with the goal of maximizing immersiveness while avoiding possible 4D sickness. They segment the film and assign appropriate 4D effects into the segments. In the editing stage, the designers create detailed 4D effects according to the production plan, which involves numerous trials and errors. Lastly, the designers evaluate the entire 4D film and carefully revise the 4D effects for a final release. On average, 4D effects production takes 16 days (2, 10, and 4 days for the three stages) by three designers for regular 4D films and 12 days (1, 10, and 1 day) by one designer for 4D rides.

Motion chairs have a number of kinematic and dynamic constraints, e.g., in the degree of freedom (DOF) and the movement range, maximum velocity, and maximum acceleration in

each independent axis. Therefore, motion effects need to be optimized to provide the best perceptual effects under the constraints of the chair. To this end, according to the interview, designers greatly rely on their contextual understanding of a film and artistic instincts, instead of using simple patterned motion effects. As such, the interviewees had difficulty in expressing their tactics and methods of motion effect design precisely in language. In our position, however, we needed rules of thumb that are adequate for automation. Hence, we inferred common design methods from the results of our motion effects analysis of the 18 4D films and presented them to the interviewees for their feedback. After long discussion, the interviewees confirmed that the following is generally accepted practice.

- Class V effects are well correlated to sound effects, e.g., those for gunfire, explosion, and engine. Hence, sound is a good source for V effects design.
- When a motion chair does not support the full six DOFs, motion in a missing direction is substituted by motion in the most similar direction.
- For class C motion effects, designers move a chair to the same direction of camera motion to align the chair's motion to viewers' viewpoint, which is critical for immersiveness. The only exception is for CF effects during acceleration or deceleration, where the chair is moved to the opposite direction to render inertia. Also, providing onset cues at precise timing is of great importance for CF effects.
- To design CS effects, designers move a chair gently and continuously by tracking the camera motion while making a full use of the chair's motion range. If the chair can no longer follow the camera motion due to the chair's motion range limit, the chair is restored to the opposite direction and then pushed again to the original direction. This slow swing motion is effective in improving presence and so is standard in CS effects.
- Class O effects are designed in a similar way to CF effects. In most cases, the movement of a character or an object of interest is transferred to the audience in the direction displayed on the screen, i.e., in the third-person viewpoint. For example, if a car

moves to right on the screen, then a chair is tilted to right, regardless of the direction to which the driver has turned the steering wheel. It is also possible that first-person experience, e.g., identifying the viewer as the car driver and tilting the chair to the direction of steering wheel rotation, is more adequate. Which method to use is decided by designers' understanding of the context.

3.3 Summary

The survey results shed light on elucidating the specific needs for synthesis algorithms of 4D effects, as well as the goals and requirements of the algorithms. Among the three stages of 4D effects production, it is evident that the editing stage can be the best beneficiary of computational algorithms; human intelligence and judgments are vital in the other two stages.

After scene breakdown, a film is partitioned to a large number of segments along with the types of 4D effects to be used in each segment. If designers can apply a synthesis algorithm to each segment and then revise the output 4D effects, it will significantly improve the productivity of 4D effects production. Our synthesis algorithms are based on this use scenario.

Among the many types of 4D effects, motion and vibration effects are most frequently used, so they must be the first aim. Among the four main classes of effects we defined, all the classes seem eligible for automatic synthesis, except class T effects that highly depend on the context. Based on these findings, we decided to develop synthesis algorithms for V, C, and O class effects. These classes cover almost all motion and vibration effects of 4D films (Fig. 3.3).

Chapter 4

Synthesis of Impulse and Vibration Effects

The goal of this algorithm, Algorithm V, is to synthesize class V motion and vibration effects that respond to the events that involve impulses and vibrations, such as gunfire, collision, explosion, and engine vibration. In this chapter, we consider mobile device as a primary target instead of 4D theater, because mobile devices are more widely used by the general public due to low cost. However, this algorithm can be used to synthesize vibration effects for 4D theater with minor modifications. We mainly present vibration rendering algorithm while briefly describing motion rendering algorithm in Section 4.5. Note that we use ‘vibrotactile’ instead of ‘vibration’ throughout this chapter to emphasize that the vibration is perceived through touch.

4.1 Translation System Overview

As shown in Fig. 4.1, our perception-level audio-to-tactile translation algorithm consists of three steps: analysis, mapping, and synthesis. In Step 1 (analysis), we extract two perceptual variables, loudness and roughness, from audio signals. These two metrics are chosen among many auditory perceptual characteristics, e.g. loudness, roughness, consonance, and brightness, for the following two reasons. First, loudness is the most important factor that

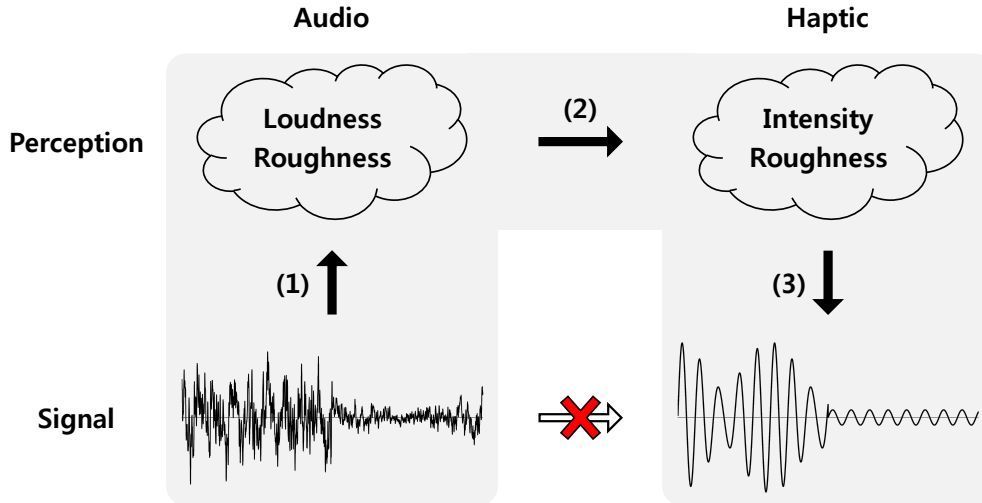


Fig. 4.1 Overview of our perception-level audio-to-tactile translation.

must be considered to determine the strength of vibrotactile feedback. Second, roughness is one of the most well-understood property in tactile perception [27, 43], and it has been used as one of the main design variables for tactile signals [8]. It is also possible to systematically modulate the roughness of vibrotactile stimuli [44, 2]. In Step 2 (mapping), the loudness and roughness of an audio signal are converted to the desired perceived intensity and roughness of a vibrotactile signal. In Step 3 (synthesis), we synthesize a vibrotactile signal with the specified perceived intensity and roughness. For this purpose, two sinusoids with different frequencies are superimposed. As will be demonstrated later, this is a simple yet effective solution for controlling the perceived intensity and roughness of vibrotactile signals. In signal-level conversion, perceptual consequences are hidden and cannot be understood without additional analysis or considerable expertise on auditory and tactile perception. In contrast, our translation framework, where cross-modal conversion is done in a higher perceptual layer, attempts to model and associate explicit perceptual variables between sound and vibrotactile stimuli. Therefore, our algorithm has a higher potential for providing well-matched, seamlessly-synchronized audio-vibrotactile pairs than the previous signal-level methods.

We implemented our perception-level translation algorithm using a smartphone (Galaxy S2; Samsung Electronics; Android platform). For vibrotactile feedback, we did not use its built-in actuator (LRA; Linear Resonance Actuator). Its bandwidth is very narrow (only a few Hz wide), which precludes rendering diverse vibrotactile stimuli. Instead, we attached a Haptuator (Tactile Labs; Canada) to the backside of the smartphone. This actuator has a wide frequency bandwidth (50–500 Hz) with excellent output linearity. The Haptuator was interfaced with the smartphone using one line of a stereo audio output (the other line to an external speaker). We calibrated the input/output relation of the Haptuator using an accelerometer attached on the smartphone. Thus, all vibration amplitudes reported hereafter represent the actual vibration strength transmitted to the hands.

4.2 Auditory Loudness and Roughness

To determine a computation model of auditory loudness to use, we tested one model by Zwicker and Fastl [95] and another model by Glasberg and Moore [20]. They are well-accepted models in auditory perception and can be applied to time-varying sound. We implemented both models on several platforms and tested their computational performance. However, none of them showed viable real-time performance. For example, faster Zwicker’s model took 700 ms on Galaxy S2 to process 4800 samples at a sampling rate of 48 kHz.

Thus, we developed a simplified auditory loudness model based on the ISO equal-loudness contours (ISO 266:2003):

$$L_a = C \sum_{f=25}^F \frac{1}{\alpha_f} 20 \log_{10} (cx_f), \quad (4.1)$$

where x_f is the signal amplitude at frequency f (Hz) and α_f is the sound pressure level (dB SPL) of the 60-phon equal loudness contour at f . c is a scaling constant depending on the data type of audio samples (1.37 for typical 16-bit integer samples). F is the highest frequency in the audio spectrum (generally 6400 Hz). C scales the entire loudness function so that L_a has the same mean as the mean computed by Glasberg’s model. C depends on F , and it is 0.065 for $F = 6400$ Hz. In theory, this approach lacks accuracy, particularly

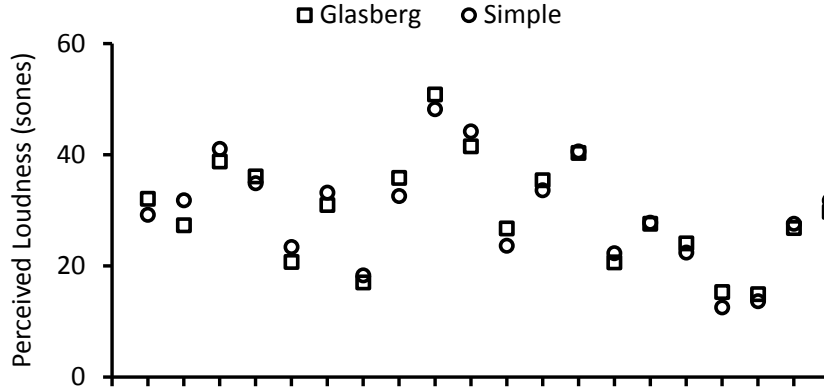


Fig. 4.2 Loudness values computed for 20 different 4096-long audio samples (sampling rate 44.1 kHz) collected from various music and movie clips. The Glasberg’s model and our simplified model (after constant scaling) produced similar results.

for time-varying sound. In practice, however, its errors are well-tolerable for short-period sounds, as demonstrated in Fig. 4.2. L_a is always computed with each 4096 samples at a 44.1 kHz sampling rate (about 93 ms long) in our system.

For auditory roughness, a number of computational models have been proposed. Examples include those by Kameoka and Kuriyagawa [35, 36], Hutchinson and Knopoff [29], and Vassilakis [81]. We use the latest Vassilakis’s roughness model, which improved several limitations of the other older models. The roughness R of an audio signal that has two frequency components with frequencies f_1 and f_2 and amplitudes x_1 and x_2 is:

$$R = \frac{(x_m x_M)^{0.1}}{2} \left(\frac{2x_m}{x_m + x_M} \right)^{3.11} \left(e^{-3.5s f_d} - e^{-5.75s f_d} \right), \quad (4.2)$$

where $x_m = \min(x_1, x_2)$, $x_M = \max(x_1, x_2)$, $f_d = |f_2 - f_1|$ and $s = 0.24 / (0.0207 \min(f_1, f_2) + 18.96)$. Then, the roughness of a complex sound can be computed by adding the R ’s of all pairs in the spectrum [81]. Our implementation finds the peaks that exceed a predefined threshold in the frequency domain, and then it computes the total roughness R_a of these peaks.

4.3 Vibrotactile Intensity and Roughness

The vast majority of tactile perception research used simple sinusoidal vibrotactile stimuli. In this case, there exist many systematic studies pertaining to the perceived intensity [82, 75] and roughness [30] of vibrotactile feedback. However, little is known about the perception of complex wideband vibrotactile stimuli. Only few studies investigated the intensity perception of vibrotactile stimuli composed of more than two spectral components (e.g., [5]). Therefore, to the best of our knowledge, synthesis of vibrotactile stimuli that satisfy the specified perceptual properties in a wide spectrum is infeasible at this moment.

Instead, we superimpose two sinusoidal vibrations with different frequencies for vibration synthesis. This method has two important merits. First, the perceptual characteristics of superimposed vibrations can be controlled in a systematic manner. This is demonstrated in the psychophysical experiments that will be described later in this section. Second, our approach is technically more viable. For example, developing miniature actuators for mobile devices that also cover low-frequency (below about 80 Hz) vibrations that convey rough, fluttering sensations is extremely challenging, because of their strict constraints on size, cost, and output magnitude. However, superposition of two sinusoids is achievable using currently available actuators, such as a piezoelectric actuator installed in several commercial smartphones and tablets, two LRAs with different resonance frequencies, or a newly patented dual-mode actuator [51].

Our translation framework uses two frequencies $f_1 = 175$ and $f_2 = 210$ Hz. f_1 was chosen as it is the resonance frequency of most LRAs (bandwidth a few Hz wide) used in mobile devices. f_2 was selected to be 1.2 times higher than f_1 , which can result in the maximum roughness when mixed with f_1 according to the recent study of Yoo et al. [90]. Both f_1 and f_2 belong to the bandwidth of recent piezoelectric actuators (150–250 Hz) adopted in tablets.

We needed concrete perceptual data as to how the amplitudes of the two different frequency components affect the perceived intensity and roughness of superimposed vibrations, but such knowledge was unavailable in the literature. Hence, we conducted two

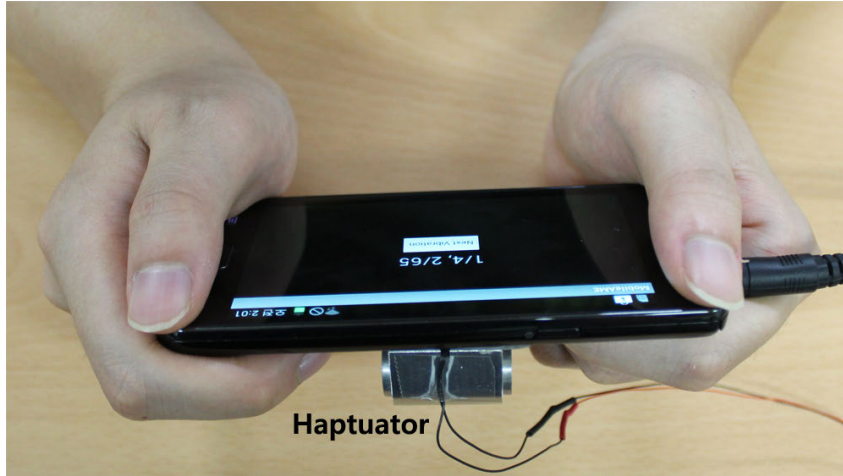


Fig. 4.3 Posture of participants used in all experiments.

psychophysical experiments to quantify the effects of vibration superposition on perceived intensity and roughness. These experiments are reported in this section.

4.3.1 Experimental Design

Twenty subjects (10 males and 10 females) participated in each experiment. Their age was 18–31 and 17–26 years in the perceived intensity and roughness experiments, respectively. All the participants reported that they had used a smartphone or tablet for more than six months and they had no known sensorimotor impairment. Each participant was paid approximately 13 USD after the experiment.

The vibrations were produced by the Haptuator attached on the back panel of Galaxy S2 ($125.3 \times 66.1 \times 8.89$ mm; 121 g). The total weight was 144.7 g. The participants held the Galaxy S2 with both hands, as shown in Fig. 4.3.

The amplitude of each frequency component was evenly divided into 10 steps between 0 and 1.1 G. All possible amplitude combinations with their sum less than 1.1 G were tested in the experiments. The total number of stimuli was 65.

On each trial, a 2-s vibration was randomly generated when the participant touched a 'Next Vibration' button displayed on the screen. Then, the participant answered its per-

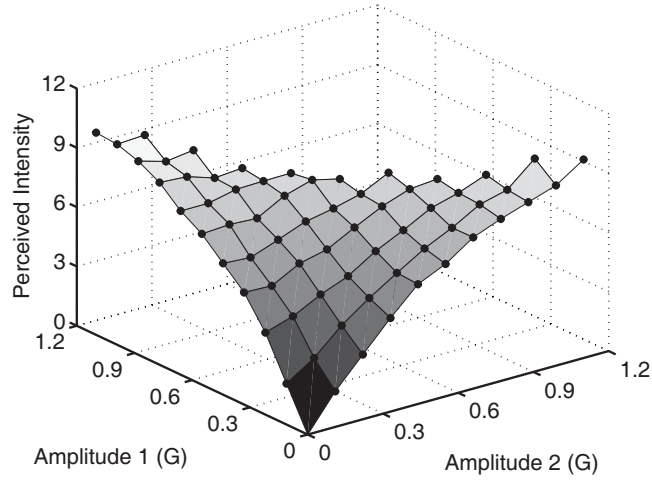
ceived intensity or roughness verbally following the standard absolute magnitude estimation procedure [19], that is, using a positive number without a reference stimulus or a modulus. The ‘Next Vibration’ button was enabled again after 8 s to prevent tactile adaptation. After evaluating all the 65 stimuli, the participant took at least 5-min rest before starting the next session. The participants repeated four sessions, which required 75 min on average. During the experiment, they wore earplugs and the headphones that played white noise to preclude any sound cues.

In each experiment, the data of the first session were excluded from data analysis, regarding the first session as training. The absolute magnitude estimates collected in the last three sessions were standardized using standard procedures [22]. The geometric mean of all responses, M_p , was calculated for each participant, and then the grand geometric mean, M_g , was computed across all stimuli and participants. Then a constant, $M_n = M_g/M_p$, was multiplied to the responses of each participant to obtain the standardized perceived intensities or roughnesses that were used for further analysis.

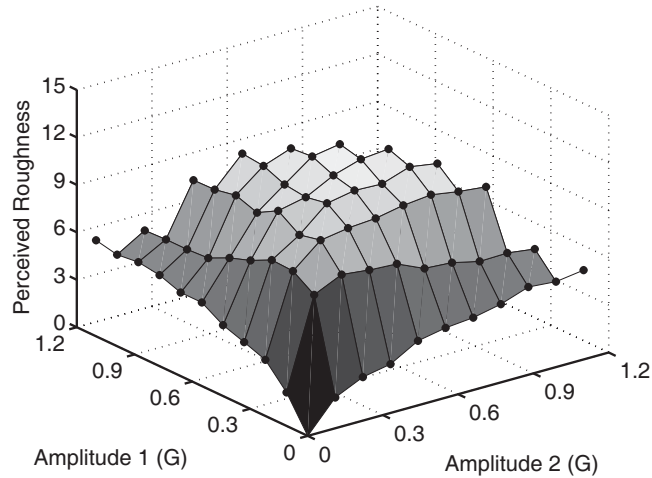
4.3.2 Results

The mean perceived intensities and roughnesses measured in the experiments are shown in Fig. 4.4. For the same total amplitude (sum of the two amplitudes a_1 and a_2), the two single-frequency vibrations (either $a_1 = 0$ or $a_2 = 0$) resulted in the greatest perceived intensities. The perceived intensity of superimposed vibrations tended to decrease as the mixing ratio became closer to 1:1. In contrast, the plot of the perceived roughnesses showed the opposite trend. For the same amplitude sum, the two single-frequency vibrations had the smallest perceived roughnesses. The perceived roughness of superimposed vibrations increased as the mixing ratio varied to even. These contradictory behaviors are advantageous for vibration synthesis, as the two perceptual variables, which have a role similar to perceptual dimensions, showed some degree of independence.

The above relations can be more clearly visualized using the total amplitude and superposition ratio. Let $A = a_1 + a_2$ be the total amplitude and $S = a_2/A$ be the superposition ratio. Using the best subset algorithm [46], we found the optimal regression models of A



(a) Perceived intensity.



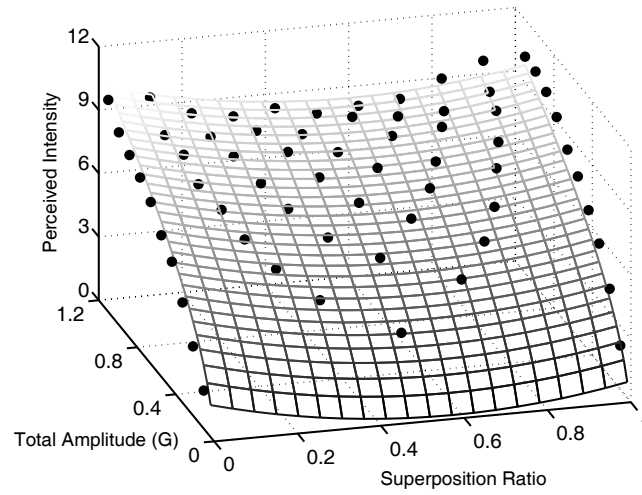
(b) Perceived roughness.

Fig. 4.4 The perceived intensities and roughnesses measured from the psychophysical experiments. Amplitude 1 and Amplitude 2 represent the amplitudes of the 175 Hz and 210 Hz components, respectively.

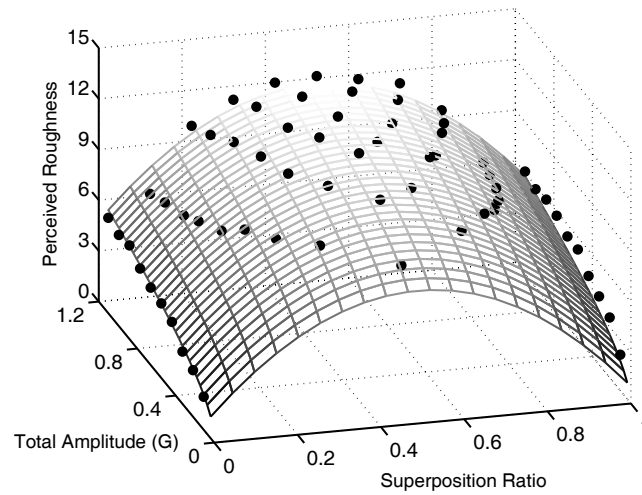
and S . The fitted functions of the perceived intensity (I_v) and roughness (R_v) were:

$$I_v = -0.521 + 10.6\sqrt{A} - 5.28S + 4.52S^2, \quad (4.3)$$

$$R_v = 0.203 + 5.63\sqrt{A} + 25.5S - 25.8S^2. \quad (4.4)$$



(a) Perceived intensity.



(b) Perceived roughness.

Fig. 4.5 Fitted perceived intensity and roughness functions. Dots represent the measured data.

The adjusted R^2 was 97.8% and 95.6%, respectively. Fig. 4.5 shows the fitted functions and original data. The aforementioned effects of the total amplitude and superposition ratio are more evident in these plots.

For our translation algorithm, inverses of I_v and R_v are necessary. The inverses solved algebraically are:

$$S = \frac{28.3 \pm \sqrt{801 - 113(R_v - 0.529I_v - 0.479)}}{56.3}, \quad (4.5)$$

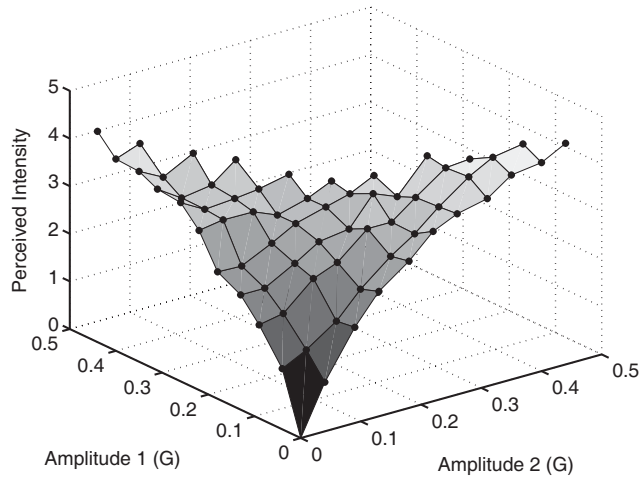
$$A = \left(\frac{25.8S^2 - 25.5S + R_v - 0.203}{3.98} \right)^2. \quad (4.6)$$

They always have a pair of solutions, but S can be imaginary numbers or lie out of the valid range (0–1). If both solutions are real and valid, then we choose the smaller S , just preferring the lower frequency vibration. If both solutions are imaginary or invalid, then we fix I_v and find the closest R_v that results in valid solutions; perceived intensity is more important for proper vibrotactile feedback. Finally, the two amplitudes are determined by $a_2 = AS$ and $a_1 = A - a_2$.

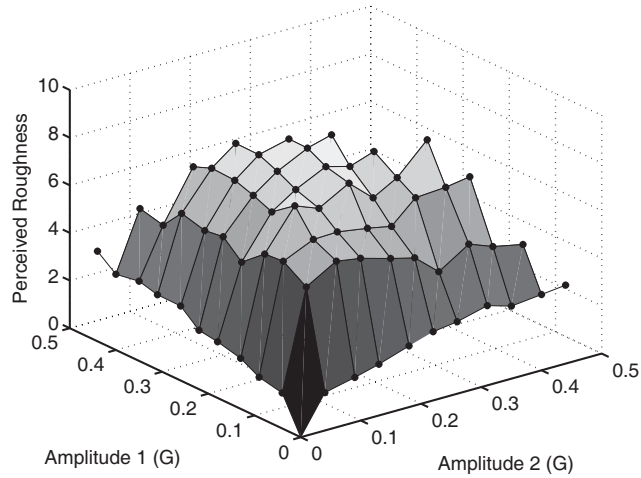
4.3.3 Discussion

The perceived intensity and roughness functions showed well-defined relationships with the total vibration amplitude and superposition ratio. For generalization of these results, the form factor of a handheld device needs to be carefully considered. The sizes of commercial smartphones are similar, and they do not incur noticeable changes to the contact conditions such as contact area and grip force. What matters more is the weight, since device weight was shown to increase the perceived intensity of tactual vibration [89]. Most contemporary smartphones weigh between 120 and 160 g, but this range is not so large as to cause drastic changes in the perceived intensity and roughness functions. For example, Yao et al. demonstrated that the perceived intensities of 110 and 130 g mockups were the same when the vibration amplitude of the 130 g mockup is about 10% smaller than that of the 110 g mockup [89]. In practice, such small amplitude differences are very difficult to detect, especially when other visual or audio stimuli are presented together as in our applications.

For further confirmation, we repeated the same experiments using a tablet (Galaxy Tab 10.1; 256.7×175.3×8.6 mm; 575 g) with five participants. The tablet was greatly larger and heavier than the smartphone used in our experiments. The results are shown in Fig. 4.6.



(a) Perceived intensity.



(b) Perceived roughness.

Fig. 4.6 Perceived intensities and roughnesses measured using a tablet.

The behaviors of the data were very similar to those in Fig. 4.4. Thus, we expect that the perceived intensity and roughness functions shown in Fig. 4.4 can be used for other devices only with a simple scaling. This can also be done within our audio-to-tactile translation models described in the next section.

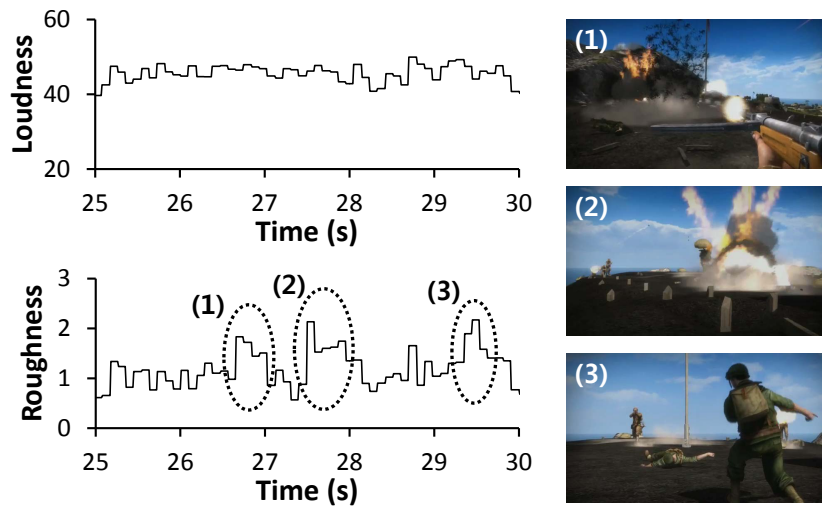


Fig. 4.7 Relationship between the auditory perceptual variables and the special sound effects accompanied in the game, *Battlefield 1943* (EA Digital Illusions CE; Sweden). Loud background music is played continuously, so the loudness L_a shows small variance. However, the roughness R_a shows the peaks (highlighted by dotted ellipses) highly correlated to the explosions and gun fires labeled by (1)–(3).

4.4 Perception Translation Model

Given the auditory and vibrotactile models of perceived intensity and roughness, the core of our perception-level translation algorithm is a conversion model from L_a and R_a to I_v and R_v . This mapping depends on application goals. As such, it can be constituted based on the designers' intuition and expertise. In our experience, the intensity model I_v is better to have a higher priority in the design because it determines whether or not a vibrotactile effect should be provided and how strong that effect should be. We designed two such models, one for games and movies and the other for music.

In the case of games and movies (especially action movies), vibrotactile feedback can play an important role in emphasizing physical impacts, e.g., explosion, gunfire, hitting, collision, and building collapse. It is also appropriate for the events or environments that people feel actual vibration, e.g., a motorcycle accelerating with loud engine sound. These are the common instances wherein most games and 4D films give manually-synchronized

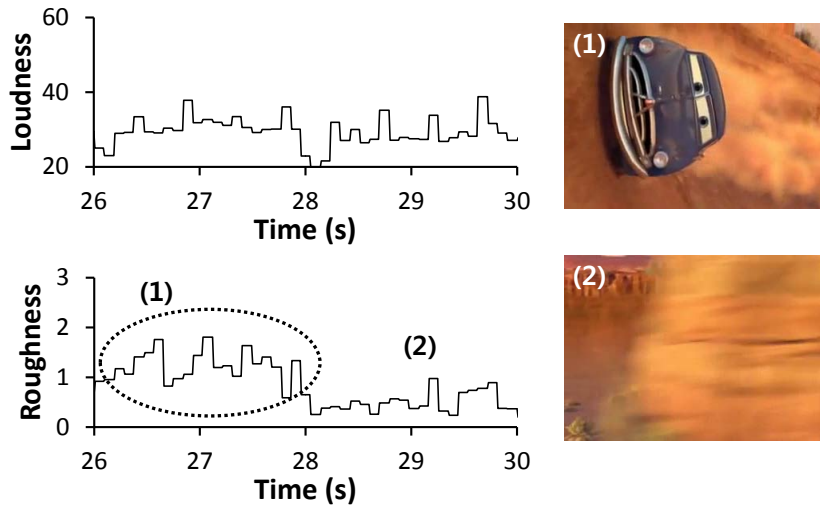


Fig. 4.8 Relationship between the auditory perceptual variables and the special sound effects accompanied in the movie, Cars (Pixar; USA). Loud engine sound is played during (1), and background music is played during (2). Even though the loudness metric cannot distinguish these two sounds, the roughness metric successfully detects only the strong engine sound.

vibrotactile effects. These events are usually accompanied with special sound effects that are perceived rougher than ambient sound or background music. Therefore, the use of the auditory roughness measure R_a , as well as the auditory loudness L_a , can contribute to detecting such instances with improved accuracy. Fig. 4.7 and 4.8 present empirical evidence supporting this conjecture.

Based on this idea, our conversion model I_v for the vibrotactile intensity is designed as:

$$I_v = c_r \sqrt{L_a} R_a^2 - o_r. \quad (4.7)$$

The square on R_a is to give emphasis on the roughness, taking into account its correlation to special sound effects. The square root on L_a deemphasizes the loudness for more accurate detection of special sound effects. c_r is a parameter analogous to volume, and o_r works as an offset that controls the onset of vibrotactile effects. The two parameters can be manually tuned by the user depending on the type of contents.

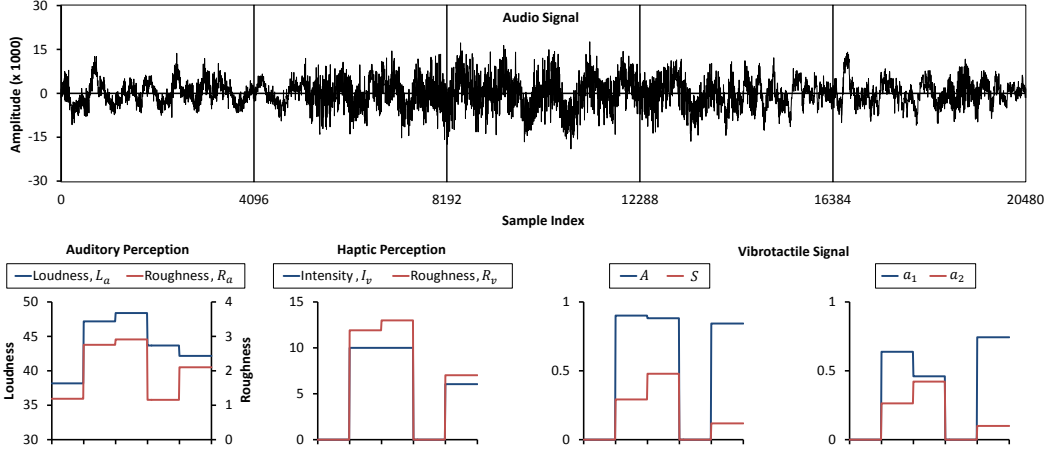


Fig. 4.9: Example results of our perception-based audio-to-vibrotactile translation algorithm.

For music, our vibrotactile intensity model is:

$$I_v = c_l L_a - o_l. \quad (4.8)$$

Here, we compute auditory loudness L_a using frequencies only below 200 Hz ($F = 200$ Hz and $C = 1.91$ in (4.1)) to lay stress upon the bass drum sound.

For vibrotactile roughness, we use the same model regardless of the content type, such that:

$$R_v = c_v R_a. \quad (4.9)$$

This simple scaling can add diverse flavors while preserving the intensity of vibrotactile effects.

The above audio-to-tactile translation models were designed based on our experiences and also by trial and error. Better models may exist depending on applications and contents.

Our current implementation of the perception-level translation algorithm computes I_v and R_v for every 4096 samples from audio files sampled at 44.1 kHz. Thus, the update rate of vibrotactile feedback is about 93 ms. This computation only takes 10 ms on average on Galaxy S2. Considering that Galaxy S2 was released in April 2011, our algorithm can run in real-time on almost all current generation mobile devices. Moreover, it does not add any

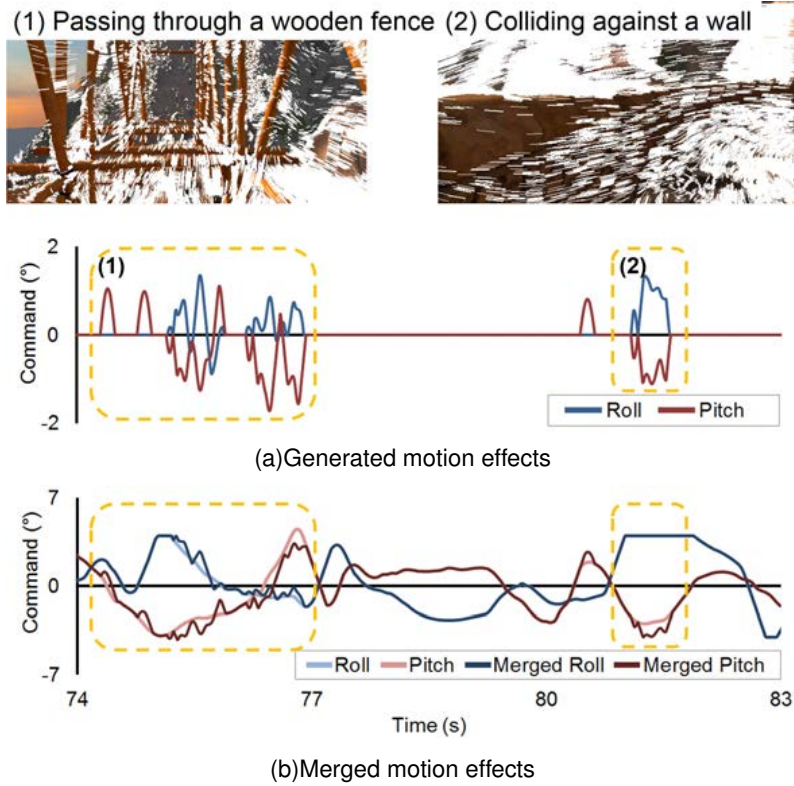


Fig. 4.10 Motion effects generated by Algorithm V (Wall of China; Simuline Inc.).

significant feedback delay considering that the feedback delay from touch to vibration is about 60 ms in recent smartphones.

Fig. 4.9 shows exemplar results of our translation algorithm.

4.5 Motion Effects Rendering for Impulse and Vibration

The desired intensity I_v of a motion effect is determined by (4.7), because motion effects are primarily used for game and movie.

The synthesis algorithm for vibrotactile effects is not applicable to motion chairs that have a greatly lower motion bandwidth. Hence, we use the following motion command tailored to motion chairs. If $I_v > 0$ for less than 0.4 s, a sine wave with a 2.5 Hz frequency (0.4 s period) is rendered for 0.2 s. If $I_v > 0$ for a longer period, irregular motion commands

are generated using the Perlin noise. These motion commands are also designed to start from and end at zero to provide the convergence property. The magnitude of these motion effects is proportional to I_v and is determined considering the workshop of a motion chair.

Motion effects rendered by Algorithm V are demonstrated in Fig. 4.10. When a tricycle breaks through a wooden fence (Fig. 4.10(1)) and hits against a wall (Fig. 4.10(2)), irregular motion effects are generated by detecting special sound effects (Fig. 4.10a). Then the effects are merged with the original motion effects, and the combined effects feel more realistic (Fig. 4.10b).

4.6 Accuracy of 4D Effect Triggering

Our translation models are designed so that they trigger vibrotactile or motion effects only when appropriate events, such as high physical impact or the appearance of vibrating objects, occur in games and movies. We tested whether our translation models work as we intended for a broad range of action scenes of game and movies. To this end, 20 game play trailers and movie clips were collected¹. Each video clip was trimmed to be 23–92 s long from action scenes. The time index of each video was manually tagged if its scene corresponded to one of the following events: explosion, gunfire, collapse, collision, hit, and shouting (by humans or animals). Then, these tagged times were compared with the initiation times of effects. The latter was defined as the time at which intensity of effect exceeds 20% of the maximum intensity.

For the 703 tagged sound effects, our perception-level translation algorithm responded correctly with a hit rate of 90.5%. The false alarm rate was 20.5% for total 800 effects. The hit rates of games and movies were similar, but the false alarm rate of movies (24.5%) was higher than for games (12.3%). Movies are usually much more challenging to handle because more diverse and complex sounds are mixed in them. The event detection performance of our algorithm for vibration feedback is among the best compared with other algorithms (e.g., [11] and Immersion’s Reverb). In addition, our algorithm has an important

¹In the last section, we discuss the game and movie types that our algorithm cannot be applied to. Such videos, e.g., games with weak sound effects, were not used in this evaluation.

advantage that no prior learning is required.

4.7 User Experiment

Lastly, we assessed the subjective performance of our perception-level audio-to-vibrotactile translation system via a user experiment. Details follow in this section.

4.7.1 Methods

Twenty-four subjects (12 males and 12 females; 19–31 years old) participated in this experiment. All participants had prior experience of using a smartphone or tablet for more than six months. No one reported known sensorimotor impairment. Each participant was paid about 13 USD.

For performance comparison, we also implemented two signal-level conversion methods. The first method sends audio signals directly to the Haptuator without any manipulation. In this case, the Haptuator’s frequency response works as a low-pass filter, and its full bandwidth is activated. This method is similar to the Crowson Tactile Motion home theater system (Crowson Technology; USA), representing an ideal case that cannot be achieved with current mobile devices. The second method is similar to the first method, but it uses a narrower bandwidth that can be implemented with recent commercial piezoelectric actuators for mobile devices. This method first finds from an audio signal the frequency with the greatest spectral amplitude. If this frequency lies outside of 25–300 Hz, no vibrotactile feedback is provided. If not, the frequency is mapped to be within 175–210 Hz linearly from 25–300 Hz. This mapped frequency is used to drive the vibrotactile actuator. Despite the similarity in the algorithm, the perceptual impressions resulted from the two methods are considerably different. The first method is much more expressive, delivering both rough fluttering and smooth vibrational sensations. The second method renders more monotone sensations that accentuate the bass band of an audio signal. We call these two conversion methods wide-band and narrow-band audio methods, respectively.

We used three types of contents, game, movie, and music, in the experiment. The parameters c_r and o_r in (4.7) were tuned to 0.035 and 0.4 for games and 0.05 and 0.4 for movies. c_l

and o_l for music in (4.8) were set to 0.1 and 3.8. For each type of contents, two videos were chosen. They were two game play trailers—Blade & Soul (MMORPG; NC Soft, Korea) and Infinity Blade 2 (iOS game; Epic Games, USA), two movies—Transformers: Revenge of the Fallen (2009, Paramount Pictures) and Fast & Furious (2009, Universal Pictures), and two music videos—Ma Boy (K-pop, Sistar) and You & I (K-pop, IU). These six videos were trimmed to be 120–138 s long. As a result, the experiment consisted of 9 conditions (3 types of contents \times 3 conversion methods), and the participants watched 18 videos (2 for each condition). For games, participants actually playing the game could be more natural. In this case, however, the occasions and the number of times vibrotactile feedback is played back cannot be controlled uniformly across the participants, so we did not choose this option.

During the experiment, the participants held Galaxy S2 with both hands, as shown in Fig. 4.3. The sound was played through an external speaker. The participant's hands and the speaker were placed on different tables to prevent vibration transmission from the speaker to the participant's hands.

Prior to the experiment, each participant filled out a questionnaire about their experience and preference of games, movies, and music. Then, the participant had a short training session in which three 55–58 s long videos, one for each content type, were presented with vibration feedback generated by each of the three conversion methods. The main experiment consisted of three sessions for games, movies, and music, respectively. The order of sessions was randomized for each participant. In each session, the participant watched two videos. Each video was presented three times consecutively with vibrotactile feedback produced by each of the three conversion methods. The orders of presenting the two videos and the three conversion methods were also randomized. After watching each video, the participant answered a questionnaire about their preference of the vibrotactile effects. After each session, the participants took at least 5 min rest to prevent fatigue and tactile adaptation. The entire experiment required around 75 min on average.

The questionnaire included the following five questions: Harmony—"Did the vibrations match to the game?"; Fun—"Did the vibrations make the game fun?"; Immersiveness—"Did

the vibrations help you be immersed in the game?"; Comfortableness– "Did the vibrations feel comfortable to enjoy the game?"; and Preference–"Did you like the vibrations played back with the game?". In the case of a movie or music, the word 'game' in the questions was replaced by 'movie' or 'music'. All questions were rated on a continuous scale by selecting a position on a horizontal line. The two ends of the horizontal line was labeled with symmetric positive and negative answers, e.g., "very uncomfortable" at the left end and "very comfortable" at the right end for comfortableness. The participants were also interviewed about the conversion methods, and these interviews were recorded.

4.7.2 Results

The experimental results are shown in Fig. 6.14. We performed one-way ANOVA on the data of each content type using the conversion method as an independent variable. The conversion method was statistically significant ($\alpha = 0.05$) for all subjective metrics except harmony and immersiveness for games. Overall, the perception-level translation showed the best scores for games, while the wide-band audio method received the best scores for movies and music. Tukey's HSD test showed that the perception-level translation was significantly better than the other two methods in comfortableness and preference for games, and the wide-band audio method was better in harmony, fun, immersiveness, and preference for movies and in fun for music.

We further classified the participants into three groups depending on their game preference. Group 1 consisted of seven participants who had played games more than five hours per week for more than three years, while Group 3 included four participants who do not play games at all. The other 13 participants were classified into Group 2. Fig. 4.12 shows that the preference of Group 1 for our perception-level translation was greatly stronger than the other groups. This result suggests that more experienced game players are better at distinguishing our event-triggered vibrotactile feedback from the other two types of signal-level feedback and that they favor event-synchronized vibrotactile effects.

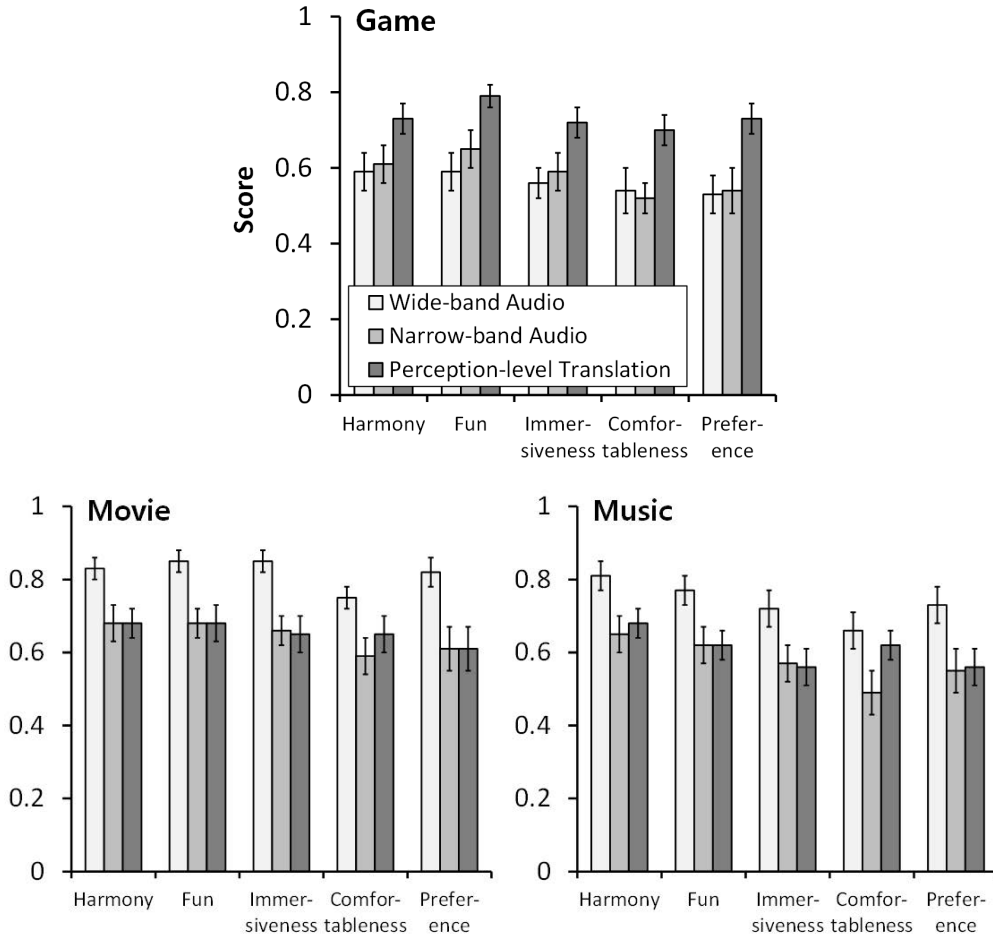


Fig. 4.11 Results of the user experiment. The error bars represent standard errors.

4.7.3 Discussion

For games, our perception-level translation algorithm is designed to feedback clear physical impacts to the player responding only to special sound effects, so that the player becomes more deeply immersed into the game. Such vibrotactile feedback can be particularly beneficial for games as game players tend to strongly identify them with the main game characters. In the interviews, 17 participants reported that vibration feedback produced by our algorithm enabled them to be more immersed into the games. In contrast, the other two

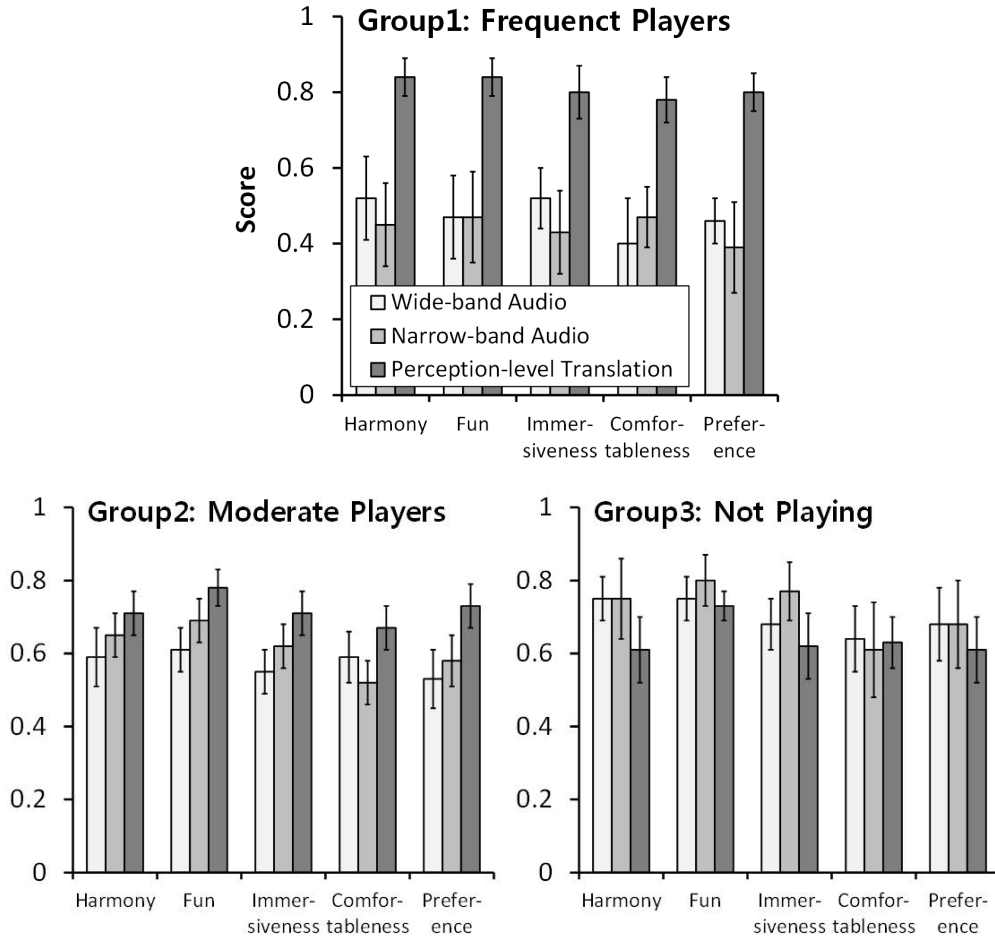


Fig. 4.12 Results of games after grouping the participants in regard to their gaming preference. The error bars represent standard errors.

signal-level methods, which generate more frequent vibration feedback invoked by both background music and special sound effects, often failed to emphasize important events. Thirteen participants said that frequent vibrations generated by the two signal-level methods were sometimes disturbing.

For movies, the wide-band audio method showed the best subjective performance. While watching movies, viewers tend to be more passive and keep a more exocentric view than

while playing games. It seems that enhancing the whole atmosphere and mood of scenes results in better movie-watching experience than emphasizing only certain events. This is advantageous for the wide-band audio method that can provide balanced vibrotactile feedback between background music and special sound effects. In addition, our translation algorithm showed a considerable false alarm rate (24.5%), as reported earlier. This could have degraded the user preference of our algorithm.

For music, many participants preferred the wide-band audio method that can present vibrations perfectly matched to music. In particular, low-frequency vibrations that have exactly the same feel to bass drum sounds were reported to be excellent for the dance music used in the evaluation.

The wide-band audio method showed much higher scores than the narrow-band audio method for movies and music, even though their timings of vibration initiations are almost identical. This demonstrates the benefit of using a wide frequency band for vibrotactile feedback. Ten subjects mentioned that the wide-band audio method gave diverse, delicate vibrations, while the narrow-band method gave relatively monotonous feedback. Note that the benefits of the wide-band audio method was enabled by the Haptuator, but such high-performance actuators are not yet feasible for mobile platforms because of size and cost.

Another important factor that must be considered in real applications is tactile adaption and fatigue. The signal-level conversion methods are apt to produce continuous vibrations, which can lead to the numbness of hands after prolonged use. Though our participants watched short video clips (less than 2 min), eight participants reported the same opinion. In contrast, our perception-level translation is relatively free from this problem as it responds to only occasional special events.

4.8 General Discussion

Two remarks need to be made. First, our translation algorithm can be easily implemented and run in real-time in a wide variety of platforms. Our current implementation runs as a separate thread, so it can be easily incorporated into any applications and products. Alternatively, our algorithm can also be embedded into the audio subsystem of an operating system

for automatic generation of vibrotactile effects with any third party applications, just like Immersion's Reverb module that is already deployed in several mobile devices. Second, our algorithm can be used with commercial wideband actuators without any modification. To use our algorithm with narrow bandwidth actuators (e.g. LRA), we can use only the vibrotactile perceived intensity converted from the auditory loudness and roughness. Then, the amplitude of vibration to render can be determined by the perceived intensity model of single sinusoidal vibrations (e.g., [75]).

Understandably, our perception-level translation algorithm has a few limitations. First, our algorithm does not work well for the contents that have weak or poor sound effects. Examples include games that play only weak sound effects (e.g., Asphalt 6; one of the popular smart phone racing games) and movies that use mostly visual effects such as slow motion or bullet time effects for emphasis. This is a limitation of all audio-based methods. Second, if background music is very rough, e.g. rock music, our translation model can fail to separate background music and special sound effects. Lastly, sounds mixed with human voice are also challenging. Generally, vibrations should not be played for conversations, but our algorithm cannot distinguish such instances since the roughness of human voice can be very high. In the future, we plan to improve the second and third limitations by using more perceptual metrics and incorporating voice removal techniques, respectively.

Chapter 5

Synthesis of Camera Class Motion Effects

5.1 Synthesis Algorithms

Our algorithms for CF and CS effects use video as a source. They include a camera motion estimation method based on the epipolar constraint between two frames (Section 5.1.1). The estimated camera motion is used to compute motion commands for both CF and CS effects, but with different motion mapping algorithms (Section 5.1.2 and 5.1.3).

5.1.1 Camera Motion Estimation

Relative camera motion between two consecutive frames is estimated using the epipolar constraint. We employ optical flow to find corresponding points and the normalized 8-point algorithm to compute a fundamental matrix. Camera motion parameters are extracted from the essential matrix [24]. The pipeline of our camera motion estimation algorithm is illustrated in Fig. 5.1.

Our camera estimation algorithm is optimized to synthesizing plausible motion effects for viewers, not to reconstructing physically-exact camera motion. It is easily implementable and widely applicable due to the use of well-established algorithms and the absence of strong assumptions (e.g, known camera intrinsics).

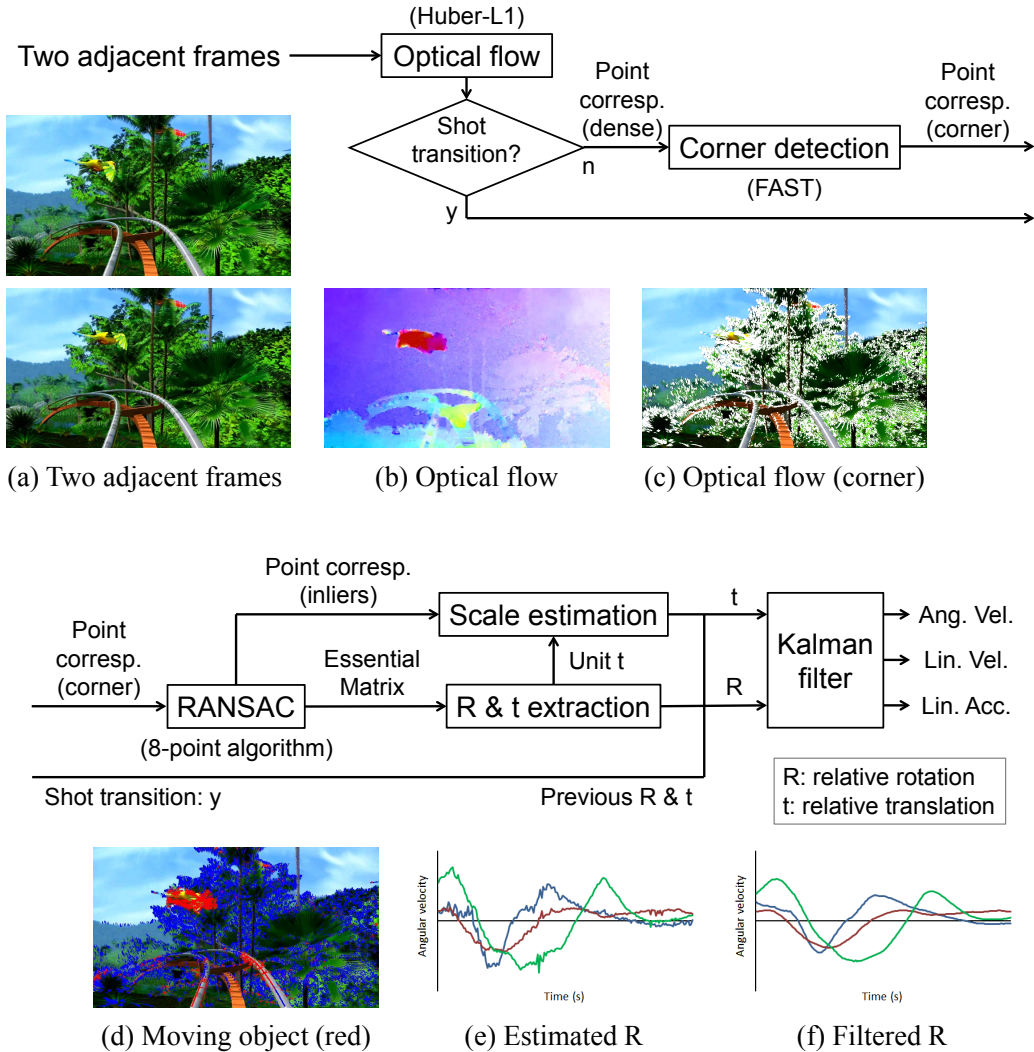


Fig. 5.1: Overview of our camera motion estimation algorithm and an example output of each step. Images (a) to (f) are example outputs taken from Amazon (ride film; Simuline Inc.). (a) to (d) are results obtained from the 607th and 608th frames. In (b), hue and brightness represent the direction and length of optical flow, respectively. In (c) and (d), flow vectors are represented by lines. In (d), blue lines represent inliers and red lines denote outliers. (e) and (f) are angular velocities estimated from the 510th to the 692th frames. Blue, red, and green lines represent angular velocities in roll, pitch, and yaw, respectively.

Identifying Corresponding Points

We employ optical flow to obtain correspondences between two frames because it typically leads to better estimation of fundamental matrix than sparse matching algorithms [55]. Although optical flow may have trouble in handling large displacement, this problem is not critical in our algorithm since corresponding points are computed only between two adjacent frames.

We use a variational optical flow estimation algorithm based on an anisotropic Huber-L1 regularization [85], called Huber-L1 optical flow hereafter, to obtain correspondence points. Huber-L1 optical flow is fast and shows reasonable performance in public benchmark [4].

For two input images I_0 and I_1 , the Huber-L1 optical flow at a point \mathbf{x} on a rectangular region $\Omega \in \mathbb{R}^2$ is defined as

$$\min_{\mathbf{u}} \left\{ \int_{\Omega} |\nabla u_1(\mathbf{x})|_{\epsilon} + |\nabla u_2(\mathbf{x})|_{\epsilon} + \lambda |\rho(\mathbf{u}(\mathbf{x}))| d\mathbf{x} \right\}, \quad (5.1)$$

where $\mathbf{u}(\mathbf{x}) = [u_1(\mathbf{x}) \ u_2(\mathbf{x})]^T$ denotes a 2D flow field. Its regularization terms are based on the Huber-L1 norms of motion gradients given by

$$|\nabla u_d|_{\epsilon} = \begin{cases} \frac{|\nabla u_d|^2}{2\epsilon} & \text{if } |\nabla u_d| \leq \epsilon \\ |\nabla u_d| - \frac{\epsilon}{2} & \text{otherwise} \end{cases}, \text{ for } \epsilon > 0. \quad (5.2)$$

The data term $|\rho(\mathbf{u}(\mathbf{x}))|$ is derived from the linearized brightness constancy constraint, such that

$$|\rho(\mathbf{u}(\mathbf{x}))| \equiv |\mathbf{u}(\mathbf{x})^T \nabla I_1(\mathbf{x}) + I_1(\mathbf{x}) - I_0(\mathbf{x})|. \quad (5.3)$$

λ controls balance between the regularization and the data term.

This Huber-L1 algorithm well preserves not only discontinuities in object boundaries but also details inside an object. The staircasing effects caused by L1 norm are alleviated by the quadratic case of Huber norm while discontinuity-preserving property of L1 norm still holds at motion boundary with $|\nabla u_d| > \epsilon$. If the mean difference of optical flows between two frames is larger than a predefined threshold, we assume that a shot transition is detected. In this case, the observed motion is disregarded, and Kalman filter is employed to smooth motion.

Optical flow estimation is often unreliable in smooth and textureless areas (e.g. sky or road), and using too many corresponding points increases processing time significantly without practical advantages in the accuracy of fundamental matrix. Therefore, we select only corner points using a high-speed corner detection algorithm, FAST [73], which rejects a large number of non-corner points very quickly (Fig. 5.1(c)). FAST classifies a point as a corner if the point is brighter or darker than the majority of its neighborhood pixels by a certain threshold. The threshold is adaptively determined in our system to maintain a sufficient number of corner points.

Computing Robust Camera Motion Parameters

Once the correspondence points between frames are obtained, the fundamental matrix is estimated by RANSAC combined with the normalized 8-point algorithm [24]. Camera motion is estimated accurately when objects in the scene are stationary. RANSAC copes with a large portion of outliers such as moving objects effectively based on the property that the corresponding points from moving objects violate the epipolar constraint, as demonstrated in Fig. 5.1(d).

To restore a camera motion, the fundamental matrix is converted to the essential matrix using the following definition:

$$\mathbf{E} = \mathbf{K}^T \mathbf{F} \mathbf{K}, \quad \mathbf{K} = \begin{pmatrix} f & 0 & w/2 \\ 0 & f & h/2 \\ 0 & 0 & 1 \end{pmatrix}, \quad (5.4)$$

where \mathbf{K} is a camera calibration matrix parameterized by image width and height (w, h) and a predefined focal length f . Since the true focal length is different from f , a restored camera motion is also different from the true camera motion by a constant scale factor. This scale difference is handled by constant gains $(c$ and $b)$ in the motion mapping step (Section 5.1.2 and 5.1.3). When the focal length changes between two frames, the restored camera motion might be different from the true camera motion beyond a constant factor. However, such distortion is negligible unless the focal length change is quite large. Moreover, the focal length change is partly absorbed into extrinsic camera parameters and then applied to synthesizing appropriate motion effects.

Without loss of generality, a camera projection matrix of the previous frame is assumed to be $\mathbf{P}_1 = [\mathbf{I}|\mathbf{0}]$. The projection matrix of the current frame $\mathbf{P}_2 = [\mathbf{R}|\mathbf{t}]$ is estimated by a singular value decomposition of the essential matrix, $\mathbf{E} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$. Then, the rotation and translation matrices are given by

$$\mathbf{R} = \mathbf{U}\mathbf{W}\mathbf{V}^T \text{ or } \mathbf{U}\mathbf{W}^T\mathbf{V}^T \text{ and } \mathbf{t} = \mathbf{u}_3 \text{ or } -\mathbf{u}_3, \quad (5.5)$$

where

$$\mathbf{W} = \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \text{ and } \mathbf{U} = [\mathbf{u}_1 \ \mathbf{u}_2 \ \mathbf{u}_3]. \quad (5.6)$$

We select the best from the possible four combinations of \mathbf{R} and \mathbf{t} in terms of reconstruction errors.

The scale s of camera translation should also be estimated since the estimated translation \mathbf{t} is a unit vector due to its inherent scale ambiguity. For a pair of corresponding points $(\mathbf{x}_i^1, \mathbf{x}_i^2)$, s is proportional to their distance $\|\mathbf{x}_i^2 - \mathbf{x}_i^1\|$ under the assumption that the 3D structure of the scene remains unchanged between two adjacent frames and there is no rotation. Based on this observation, \mathbf{x}_i^1 in $\|\mathbf{x}_i^2 - \mathbf{x}_i^1\|$ is replaced by $[\mathbf{R}|\mathbf{0}]\mathbf{X}_i$ to cancel out the distance originated from the rotation of camera, where \mathbf{X}_i is a 3D point triangulated from $(\mathbf{x}_i^1, \mathbf{x}_i^2)$. Therefore, the scale of translation can be approximated by

$$s \approx \frac{\sum_i^N \|\mathbf{x}_i^2 - [\mathbf{R}|\mathbf{0}]\mathbf{X}_i\|}{N}, \quad (5.7)$$

where N is the number of reconstructed points. This provides sufficiently accurate solutions for our application unless scene changes are very abrupt.

Once the relative geometric configuration of camera between two frames is determined, it is straightforward to compute camera motion parameters—angular velocity, linear velocity, and linear acceleration, which are smoothed by Kalman filter (Fig. 5.1(f)). These output variables are used for the synthesis of CF and CS effects.

5.1.2 Synthesis of Fast Camera Motion Effects

Our synthesis algorithm for CF effects, Algorithm CF, includes the camera motion estimation and a motion command synthesis. For description, ω , v , and a represent angular

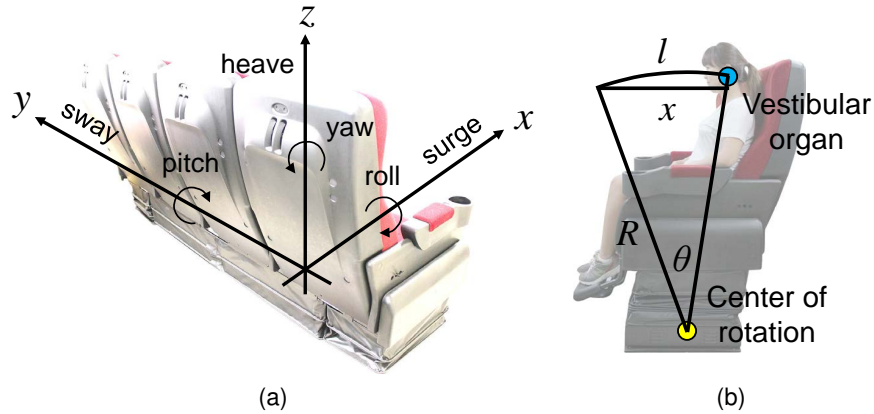


Fig. 5.2 (a) Definition of 6-DOF motions. (b) Linear-to-angular approximation (surge to pitch). For small θ , $x \approx l = R\theta$, so $x \propto \theta$.

velocity, linear velocity, and linear acceleration, respectively. Subscripts, r , p , y , g , s , and h denote motion in roll, pitch, yaw, surge, sway, and heave, respectively, of a motion chair, as shown in Fig. 5.2a. $\mathbf{p} = [p_r, p_p, p_h]^T$ is the final command to a motion chair. Our motion mapping method for Algorithm CF is depicted in a block diagram shown in Fig. 6.5.

For fast camera motion, we provide *vestibular feedback* to improve 4D experience. The human vestibular system responds to only acceleration, so not to linear velocity, but can also sense angular velocity through the linear acceleration induced by the centrifugal force [21]. Thus, we use the three angular velocities and three linear accelerations of the camera as the input.

Physically exact reconstruction of camera motion using a motion chair is impossible due to two constraints: a motion chair has a limited workspace and may not support the full six DOFs. For the former, the motion chair needs to return to its origin after generating one effect to be ready for the next motion. For the latter, motion in the unsupported directions should be replaced by motion in the available directions. We handle these two requirements as follows.

For the first requirement, we use a washout filter, a high-pass filter that has a proven convergence property to the initial state [60]. Its high-pass filtering renders high-frequency

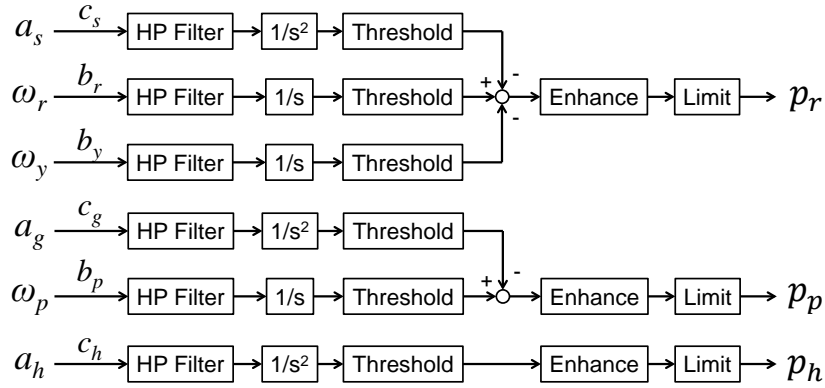


Fig. 5.3 Motion synthesis algorithm for CF effects.

onset cues while removing the low-frequency energy in motion that tend to push a motion chair to its workspace limit. The washout filter is a de facto standard in motion simulators. We use a first-order Butterworth filter with a cutoff frequency of 1.0 Hz for ω and a second-order Butterworth filter with a cutoff at 2.5 Hz for a .

It should be clarified that we do not consider tilt coordination, a technique for simulating sustained acceleration, such as gravity and centrifugal force, by tilting a motion chair for a relative long time. Tilt coordination is included in the original washout filter and commonly used for motion simulators with training purposes. Our decision is due to several reasons. First, tilt coordination is generally implemented using low-pass filters, but this technique is not very effective with general motion chairs for 4D films because of their much smaller workspace (8-14 degrees) than those of flight or driving simulators (usually greater than 40 degrees). Second, if sustained acceleration is provided with more abrupt CF effects, it is generally masked and not clearly perceptible to viewers. Third, we did not find instances in which only sustained acceleration was predominant out of the 2278 4D effects we analyzed (Section 3). Lastly, including tilt coordination makes motion synthesis algorithms and associated gain tuning more complicated.

For the second requirement, we use the motion substitution rule shown in Fig. 6.5: (roll, yaw, sway) of camera \rightarrow roll of motion chair, (pitch, surge) \rightarrow pitch, and heave \rightarrow heave.

This mapping is tailored to 3 DOF motion chairs in 4D theaters, which typically support only roll, pitch, and heave motions [86]. Expert designers also use the same or similar ways (Section 3.2).

A linear motion is approximated by an angular motion at the corresponding axis, as illustrated in Fig. 5.2b. Therefore, increasing sway corresponds to decreasing roll (Fig. 5.2a), so sway motion is subtracted from roll motion (Fig. 6.5). Increasing surge corresponds to increasing pitch (Fig. 5.2a). However, we subtract surge motion from pitch motion, instead of adding it to (Fig. 6.5), to render inertia induced by acceleration/deceleration in the viewer's frontal direction. Note that this technique responds to only high-frequency acceleration/deceleration, unlike tilt coordination that renders low-frequency sustained acceleration/deceleration. Yaw angle is linked to roll (Fig. 6.5) as a rotation of a volumetric object in yaw incurs a linear change in sway, which is mapped to roll in our substitution rule.

Scale factors b and c in Fig. 6.5 take care of the unit and range differences between ω and a . They also work as gains for motion effects. Soft thresholding is applied to remove small noise during the stationary state.

The combined motion commands for roll, pitch, and heave can be enhanced for more dynamic effects by the following nonlinear amplification: for a motion command m ,

$$\hat{m} = \text{sgn}(m) \alpha \left(\frac{\|m\|}{\alpha} \right)^\beta, \quad (0 < \beta \leq 1) \quad (5.8)$$

where \hat{m} is an enhanced motion command, α is the maximum displacement or angle of that motion, and β is a gain for motion enhancement. Decreasing β makes motion effects more dynamic and fun deviating from the estimated camera motion, while using $\beta = 1$ renders the original motion as it is. According to our experience, the range of β for perceptually-best motion effects is 0.7–0.9 for most 4D films.

As the last step, we limit the motion commands not to exceed the maximum displacement and velocity of the motion chair for safety.

5.1.3 Synthesis of Slow Camera Motion Effects

The goal of our synthesis algorithm for CS effects, Algorithm CS, is to re-create slow camera motion using a motion chair to provide viewers with an illusion of observing the scene from a distance while slowly moving or floating at the camera position within the space of the scene. To this end, Algorithm CS transforms continuous camera motion, possibly to one direction for an extended period of time, to swinging chair motion staying within the workspace of the chair, as is done by 4D effects designers (Section 3.2).

After camera motion estimation, Algorithm CS proceeds in two phases. In the first phase shown in Fig. 5.4a, a 3D camera trajectory \mathbf{p}^1 is computed from the linear and angular velocities of camera motion. Using velocities as input enables constant velocity motion, which is not possible with the washout filters for Algorithm CF in the linear directions. Here, surge, sway, and yaw camera motions are substituted by pitch and roll motions as in Algorithm CF. Surge motion is now added to pitch motion as there is no need for inertia rendering in CS effects. Then \mathbf{p}^1 is obtained by integration. \mathbf{p}^1 is computed for all frames.

Unlike washout filters, the operations in the first phase do not have the convergence property to the initial state. Thus, \mathbf{p}^1 may exceed the workspace of the motion chair. We handle this problem by mimicking the 4D designers' strategy in the second phase. For this purpose, \mathbf{p}^1 is further segmented and scaled as shown in Fig. 5.4b. \mathbf{p}^1 is divided to many segments using either equi-time or equi-distance segmentation. In the equi-time segmentation, all motion segments have the same duration. In the equi-distance segmentation, all motion segments are enclosed within a sphere of the same radius. In the first and last segments, the motion chair should start from and come back to the origin, respectively, so only a half of the workspace is usable. They are made by using a half duration or a half radius of the enclosing sphere compared to the other segments.

Then, each segment of \mathbf{p}^1 is scaled to fit into the workspace of the motion chair. For $\mathbf{p}^1(i)$ ($i_{k-1} \leq i \leq i_k$) in the k -th segment, the scaled motion command is computed by

$$\mathbf{p}^2(i) = \frac{\mathbf{p}^1(i) - \mathbf{c}^*}{\max_{i \in [i_{k-1}, i_k]} \|\mathbf{p}^1(i) - \mathbf{c}^*\|} \circ \boldsymbol{\alpha}, \quad (5.9)$$

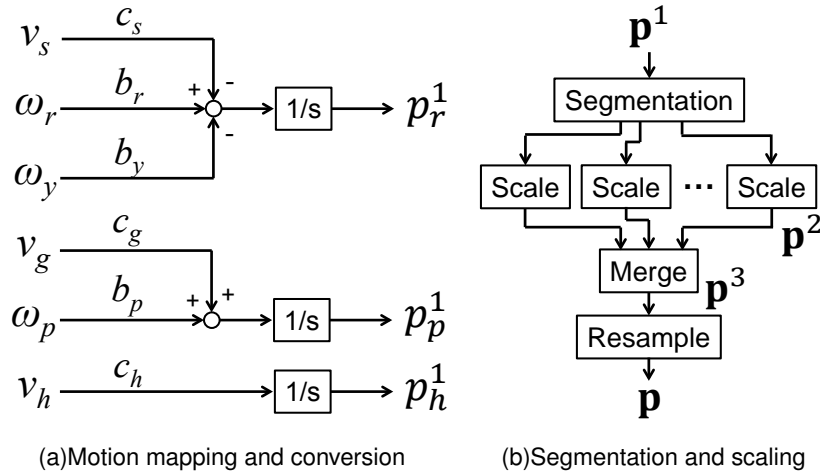


Fig. 5.4 Motion synthesis algorithm for CS effects.

where $\alpha = [\alpha_r, \alpha_p, \alpha_h]^T$ is the maximum motion range and \circ denotes the element-wise product operator, and

$$\mathbf{c}^* = \begin{cases} \mathbf{p}^1(1) & \text{the first segment} \\ \mathbf{p}^1(N) & \text{the last segment} \\ \underset{\mathbf{c}}{\operatorname{argmin}} \left(\max_{i \in [i_{k-1}, i_k]} \|\mathbf{p}^1(i) - \mathbf{c}\| \right) & \text{otherwise} \end{cases}, \quad (5.10)$$

where N is the total number of elements in \mathbf{p}^1 . The last case of (5.10) computes the center of the minimum enclosing sphere of the segment.

All scaled segments of \mathbf{p}^2 are linked together to \mathbf{p}^3 by inserting linearly interpolated points between adjacent segments. C^1 discontinuities in \mathbf{p}^3 are not perceptually salient owing to slow chair motion. \mathbf{p}^3 is then resampled to ensure that the final motion command \mathbf{p} has the same duration as \mathbf{p}^1 before segmentation. An example demonstrating the motion synthesis of Algorithm CS is given in Fig. 5.7.

Since our motion scaling is distance-to-distance, the equi-distance segmentation preserves the camera velocity more accurately. In the equi-time segmentation, the distance scale from camera to motion chair varies segment by segment, providing continuously moving sensations with less velocity variations. 4D effects designers can choose one of the two

Table 5.1 Parameter values for FlowLib v3.0.

Parameter	Value
model	INTERPOLATE_FAST_HL1
iters	5
warps	2
scale_factor	0.9

Table 5.2 Default b and c for Algorithm CF and CS.

Algorithm	b_r	b_p	b_y	c_g	c_s	c_h
CF	1800	2200	1800	150	120	120
CS	400	400	400	1	1	0.1

segmentation methods suitable to their needs.

It is noted that Algorithm CS results in fundamentally different motion effects from Algorithm CF, even when scene changes are relatively slow. For example, the camera moves in one direction at a constant velocity, Algorithm CS moves the chair in the same direction at the same velocity (though scaled), but Algorithm CF makes zero output due to zero accelerations.

5.1.4 Default Parameters and Implementation Issues

We present the default parameters of our motion effects synthesis algorithms to help reproduction by readers. All motion effects used in the experiments reported in Section 5.2 and 5.3 were generated using the default parameters.

Most parameters for camera motion estimation can be used without tuning. We tested FlowLib v3.0 [85] and Liu’s code [53] for optical flow estimation. Although both implementations with their default parameters synthesized good motion effects, we tuned several parameters of FlowLib v3.0 to speed up computation as shown in Table 5.1. All input images were scaled to 640-pixel wide, and the focal length f was fixed to 640.

Scale factors b and c for Algorithm CF (Fig. 6.5) and Algorithm CS (Fig. 5.4a) are summarized in Table 5.2. In Algorithm CF, the b and c gains are tuned while considering the workspace volume. In their default values, b_p and c_g are larger than the other corresponding gains to utilize the larger limit of pitch of our motion chair (pitch: ± 7 degrees; roll: ± 4

degrees; heave: ± 4 cm). Thresholds for noise removal in the steady state were 0.05 for angular velocities and 0.15 for linear accelerations. β is the single design parameter of Algorithm CF that 4D designers control to make motion effects more dynamic (smaller β) or less dynamic (larger β). Its default value is 0.8.

In Algorithm CS, 4D designers determine the segmentation method and the length of segments for each scene. For example, we used the equi-time segmentation in all the experiments we report, and the length of segments varied from 10 to 35 s. The default gains of b and those of c should be the same, respectively, since the motion trajectories of Algorithm CS already make use of the full workspace. The only exception is c_h for heave motion. We use a much smaller value to reflect the fact that human sensitivity to heave motion is much inferior to that for surge or sway, but using high c_h decreases the usable range of roll and pitch motion in our motion chair. This is particularly important for slow motion.

In Algorithm V, the scaling constant c was 0.05, and the offset o was 2.0.

For motion chairs with a higher DOF, the motion substitution rule described in Section 5.1.2 needs to be revised for the additional axes, but it is generally straightforward. For example, for a 4-DOF chair with roll, pitch, yaw, and heave, the yaw velocity ω_y in Fig. 6.5 is no longer fed to roll motion p_r , but to yaw motion exclusively.

5.2 Results

We present sample results for each algorithm in this section, followed by two user studies in the next section.

5.2.1 Camera Motion Estimation and Motion Effects

Fig. 5.5 presents the results of camera motion estimation and CF effects synthesis using a video we recorded in a car driving on a paved city road. The angular velocities and linear accelerations of the camera were measured using an additional motion sensor (CruizCore XA3300; Microinfinity Corp.) for ground truth. In Fig. 5.5(a), it is clear that our camera motion estimates track the measured angular velocities of the camera with high accuracy and almost no delay. In Fig. 5.5(b), the estimated linear accelerations well reflect the gen-

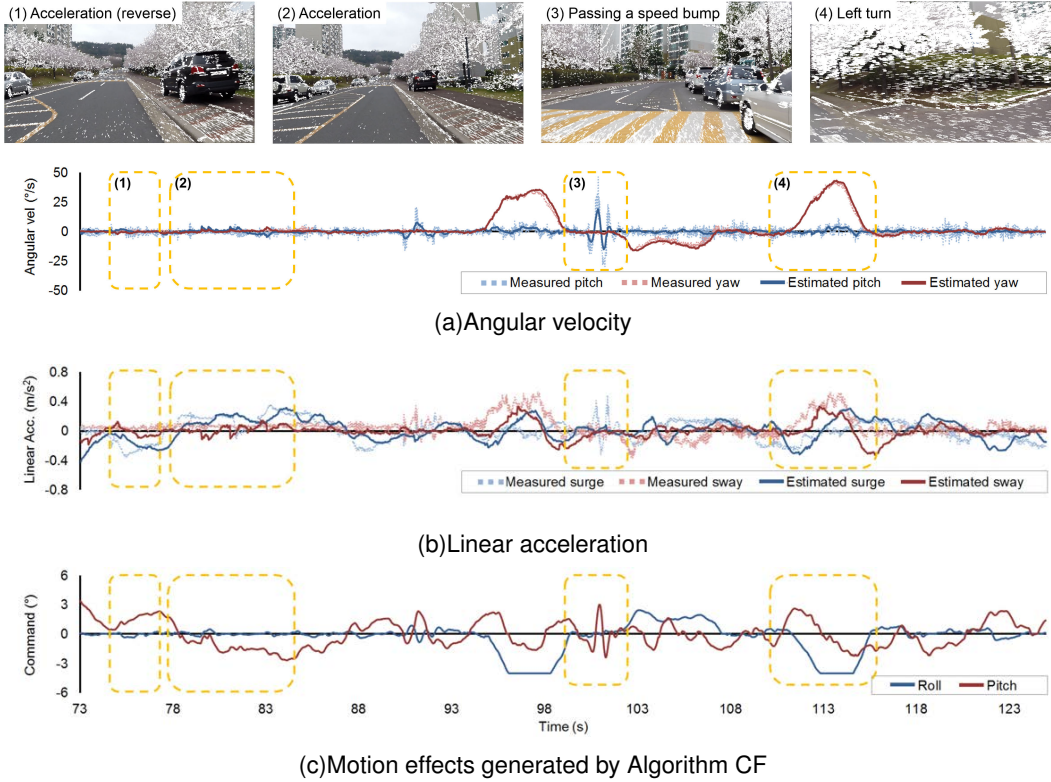


Fig. 5.5: A comparison between measured and estimated camera motions (a, b) and synthesized CF motion effects (c). Highlighted parts (1)–(4) are also shown in upper images. Note that the high-frequency noise in the measured camera motion is injected from ambient noise in the car.

eral trends of the measured linear accelerations, but with some errors and delays. This is expected since camera translation estimation is inherently less accurate than rotation estimation [72] and acceleration estimation requires differentiation that amplifies noise. This is one reason behind the design of our motion synthesis algorithms that use all angular velocities and linear accelerations for robust motion synthesis. Synthesized motion commands in Fig. 5.5(c) express the camera motion faithfully: (1) car acceleration (on the reverse) \rightarrow chair pitch increase (leaning forward), (2) car acceleration \rightarrow chair pitch decrease (leaning backward), (3) car passing a speed bump \rightarrow an abrupt and short oscillation in chair pitch, and (4) car left turn \rightarrow chair roll decrease (tilting left), and car deceleration before a curve

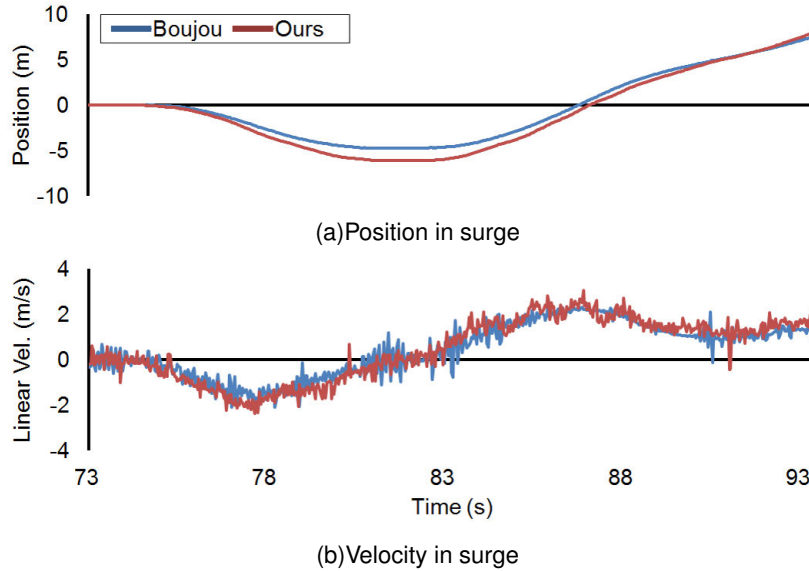


Fig. 5.6 Comparison of surge motions estimated by our camera motion estimation algorithm and Boujou 5 before smoothing or filtering. The same video used in Fig. 5.5 was used as input.

→ chair pitch increase and car acceleration after the curve → chair pitch decrease.

We further investigated the accuracy of our camera motion estimation algorithm by comparing it with Boujou 5 that was used in [79]. This comparison focused on surge since it is the most difficult DOF to estimate using only monocular images. Recall that whereas the output of our algorithm is local velocities, that of Boujou is global positions (after full 3D trajectory reconstruction). For fair comparison, we used the first 20 seconds of the driving video (Fig. 5.5) on the straight lane before a large orientation change. The velocities estimated by our algorithm were converted to positions using the cumulative trapezoidal numerical integration, and the positions computed by Boujou were converted to velocities using the symmetric derivative. Results are provided in Fig. 5.6, where the two methods showed very similar position and velocity profiles of surge. This is expected to some extent since Boujou in its first step includes essentially the same procedure of camera pose estimation in the local coordinate frame [24, 71].

Fig. 5.7 illustrates an example result of Algorithm CS. The motion effects are generated

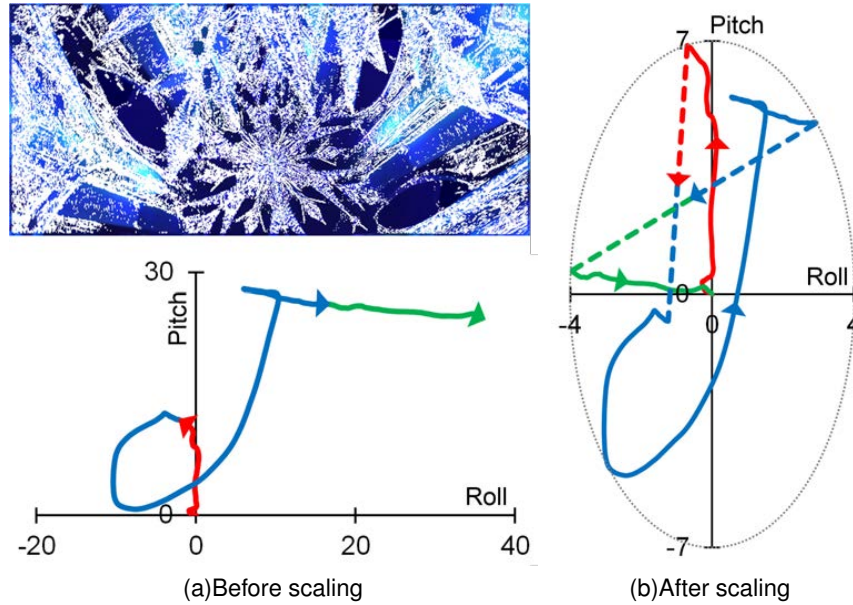


Fig. 5.7 Motion effects generated with Frozen by Algorithm CS with the equi-time segmentation. Red, blue, and green lines are for the first, second, and third segments, respectively. In (b), dashed lines are interpolated motions to connect the segments. A dotted ellipse represents the workspace of our motion chair.

for a scene in Frozen in which Elsa builds an ice castle. The motion chair is tilted forward slowly when the camera approaches to the castle (a red segment in Fig. 5.7). Then, the chair is tilted backward because the camera faces up to the ceiling, and then it is tilted forward again when the camera looks down on the floor (a blue segment). The left and right roll motions describe the counter-clockwise and clockwise rotations of the camera, respectively. In this scene, the focal length also changes due to zooming. Since we used a fixed focal length model (see \mathbf{K} in Section 5.1.1), the zooming is replaced by forward or backward translation, which well matches zoom-in or zoom-out, respectively, in practice. As shown in Fig. 5.7b, the CS motion effects fully utilize the workspace of the motion chair.

Table 5.3 Execution times (ms) for one frame.

# of corner points	< 5000	< 15000	< 30000
Optical flow		103.892	
Cut detection		0.131	
Corner detection		0.503	
RANSAC	104.653	173.521	328.953
R&t extraction	2.196	11.701	29.381
Kalman filter		0.013	
CF motion mapping		0.032	
CS motion mapping		0.048	
Algorithm V		0.468	

5.2.2 Computational Complexity

We measured the computational complexity of our synthesis algorithms on a commodity PC (2nd generation Intel i5-2400 3.10 GHz CPU, 4.00 GB RAM, and NVIDIA Geforce GTX 760 graphics card), and the results are summarized in Table 5.3. All input frames were resized to 640-pixel wide, and optical flow is computed on NVIDIA CUDA. Most steps take constant times, except RANSAC and the computation of \mathbf{R} and \mathbf{t} in Section 5.1.1. The computational times of the latter two depend on the number of corner points.

The number of corner points can be adjusted by a threshold (Section 5.1.1). In our implementation, using 10,000 corner points allows robust camera motion estimation. In this case, our algorithms achieve approximately 4 frame/s, which is sufficient for our use scenario (Section 3.3). For example, for a film that has 24 frames per second, it takes only 3 min to synthesize a motion effect for a 30-s scene (usual motion effects are greatly shorter). A better use strategy is to execute the camera motion estimation that requires expensive computations during the scene breakdown stage. Then in the editing stage the designers can generate various motion effects using the motion mapping algorithms almost instantly (Table 5.3).

5.3 User Experiments

We carried out two user experiments to assess the perceptual quality of our motion effect synthesis algorithms, one with general users and the other with 4D experts.

5.3.1 Experiment I: General Users

The goal of Experiment I was to compare the immediate benefits that general users perceive from three different sets of motion effects: those randomly generated, synthesized by our algorithms, and manually designed by 4D experts.

Methods

The participants were eighteen students (13 male and 5 female; 18–32 years old) recruited from the authors' institution. None of them reported prior involvement in 4D effects production. Some participants had limited experiences of watching 4D films, eight times at most. Each participant was paid 10 USD after the experiment.

The motion chair (4DX; CJ 4DPLEX) used in the experiment had 3 DOFs for roll ($\pm 4^\circ$), pitch ($\pm 7^\circ$), and heave (± 4 cm). 3D images were projected onto a 94-inch screen by an EB-W16SK polarized 3D projector (Epson corp.). The participants wore passive 3D glasses during the experiment. Sound was played through NS-150 5-channel home theater speakers (Yamaha corp.).

Three 4D films were used in the experiment: Wall of China, Fairy Balloon Ride, and Frozen. The first two were ride films with running times of 3'59'' and 4'12''. For Frozen, a 3'36'' clip in which the heroine Elsa sings the main song "Let it go" was used.

The motion effects for Wall of China and Fairy Balloon Ride were synthesized using Algorithm CF. Algorithm V was also used for Wall of China. The motion effects for Frozen were generated by Algorithm CS. These automatic motion effects (AE) were compared with two other conditions, randomly-generated effects (RE) and manually-designed effects (ME). RE were made using Perlin noise so that they had similar strength and frequency to ME. RE served as a baseline in the experiment. ME were commercial ones that had been played at 4D attractions or 4D theaters, purchased from Simuline or CJ 4DPLEX. It

is noted that the manual effects for Frozen included class O motion effects, as well as class C effects. All regular 4D films include both classes of motion effects, and we were unable to attain those with only class C effects. This is in contrast to AE for Frozen that included only CS effects.

The experiment consisted of three sessions, one for each 4D film. In each session, one 4D film was presented three times with different sets of motion effects. The order of the 4D films and the motion effect sets was randomized for each participant. After each session, the participant took at least 5-min rest to prevent motion sickness and fatigue. The participant finished the experiment in about 1 hour.

After watching each 4D film sitting in the 4D chair, the participants answered a questionnaire. It included the following four questions: Harmony—“Did the motion effects match to the film?”; Immersiveness—“Did the motion effects help you be immersed in the film?”; Fatigue—“Did the motion effects make you feel tired?”; and Preference—“Did you like the motion effects provided with the film?”. All the questions were rated on a continuous scale by selecting a position on a horizontal line. The two ends of the horizontal lines were labeled with symmetric positive and negative answers; for example, “very inharmonious” at the left end and “very harmonious” at the right end for harmony.

Results

Experimental results are shown in Fig. 5.8. We performed one-way ANOVA on each question using motion effect set as an independent variable for each 4D film. Motion effect set was statistically significant ($p < 0.05$) for all the subjective metrics and all the 4D films. Tukey’s HSD tests were then used for post-hoc multiple comparisons, and their results are also represented in Fig. 5.8.

ME showed the best scores in all the metrics and the conditions, except fatigue for Frozen. AE generally received comparable scores to ME. Significant differences were observed in only fatigue for Wall of China and Frozen. In fact, ME felt more tiring than AE for Frozen. It is presumably because ME presented much more motion effects than AE in that particular condition. As mentioned earlier, ME for Frozen included both C and O

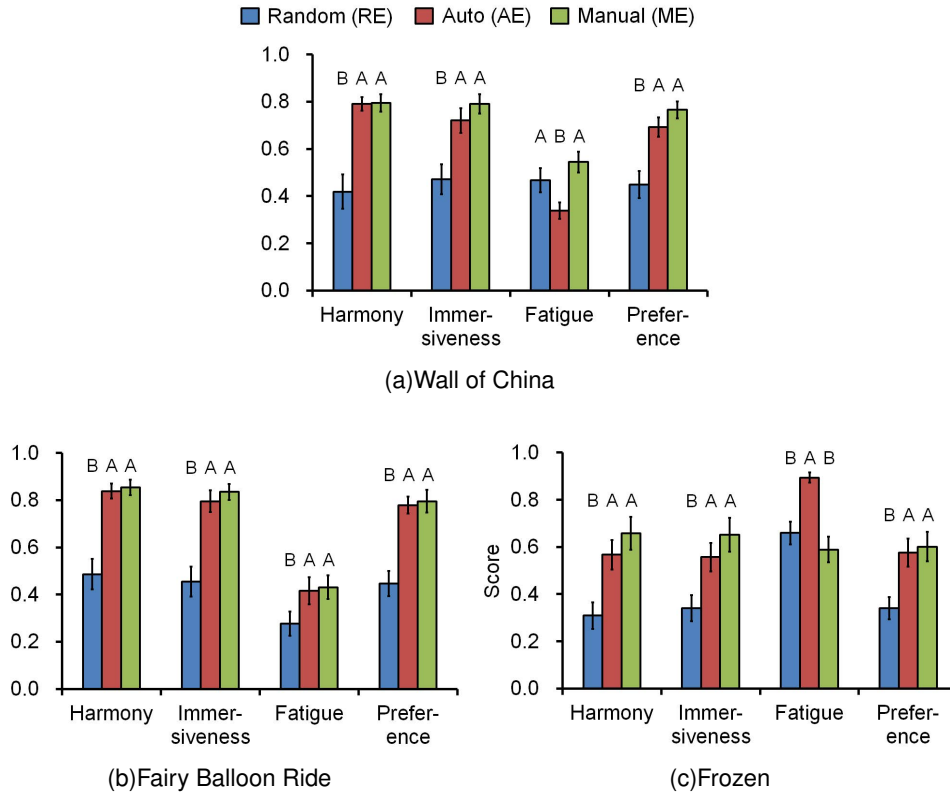


Fig. 5.8 Results of Experiment I with general users. Error bars represent standard errors. Note that the scale for fatigue is inverted so that the higher the fatigue score, the better the performance (less tiring), to be consistent with the others. The sets of motions effects marked with the same alphabets indicate that they did not show statistically significant differences by Tukey's HSD test.

motion effects, while AE provided only CS effects. Between AE and RE, AE obtained significantly higher scores in all the metrics and the conditions, with one exception in fatigue for Wall of China.

To gain further insights, Fig. 5.9 presents exemplar motion effects of RE, AE, and ME alongside the dominant camera motions annotated by a human viewer. It can be seen that the roll and pitch motion commands of RE are not correlated with the dominant camera motion, but those of both AE and ME express the dominant camera motion effectively. For example, left camera turn (labeled by L) led to decreasing chair roll (tilting left), right

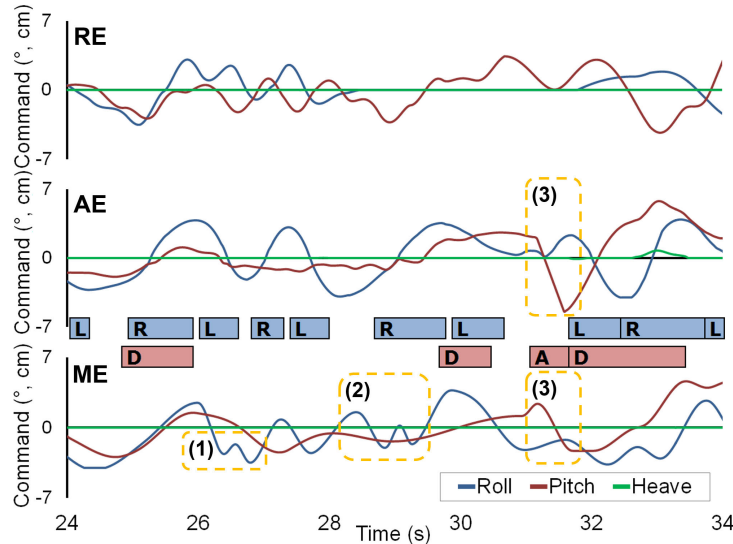


Fig. 5.9 Examples of randomly-generated (RE), automatic (AE), and manually-designed (ME) motion effects (Fairy Balloon Ride; a ride film from Simuline Inc.). Blue and red boxes represent the dominant camera motions that were manually annotated by a human viewer. The blue boxes indicate horizontal motions (L: left turn and R: right turn; related to roll), and the red boxes are for vertical motions (D: descent and A: ascent; related to pitch).

camera turn (R) to increasing chair roll (tilting right), camera descent (D) to increasing chair pitch (leaning forward), and camera ascent (A) to decreasing chair pitch (leaning backward). There are also some specific differences between AE and ME. For example, ME includes two small oscillations (Fig. 5.9(1) and (2)) inserted to express the impact caused by collisions to a snowball in the movie. Such effects are not generated by AE since Algorithm V is not able to detect collisions of weak sound effects. In addition, ME is generally smoother than AE. An obvious example is marked in Fig. 5.9(3); it appears that the expert designers made motion effects for the sudden ascent of a balloon much smoother than its actual value, probably for the concern of motion sickness.

In conclusion, in terms of perceptual quality to general users, the motion effects synthesized by our algorithms outperformed the random effects by large extent, and were comparable to the commercial manual effects, at least for the tested 4D films.

5.3.2 Experiment II: 4D Experts

Inspired by the promising results of Experiment I, Experiment II aimed at gauging the quality of motion effects synthesized by our algorithms with the careful eyes of 4D experts.

Methods

Twenty-three 4D experts (19 male and 4 female; 26–43 years old) who were working in CJ 4DPLEX participated voluntarily in this evaluation. Ten of them were responsible for 4D effects production, nine of them for software or hardware development, and the other four for planning. On average, the participants had 34 months of work experience in their current position. All of them had ample experience in making or evaluating 4D effects.

In addition to the three 4D films used in Experiment I, two other 4D films, *The Hobbit: The Desolation of Smaug* (45'') when Gandalf climbed a mountain to meet Radagast) and *Amazon* (4'11'') were used in this experiment. Algorithm CS was used to generate motion effects for the *Hobbit*, and Algorithm CF was used for *Amazon*. The motion effects for the other three 4D films were the same as those used in Experiment I.

Since our expert participants could distinguish details in motion effects, they were very likely to notice whether motions effects were manually designed or automatically synthesized. This can instill a substantial bias to blind comparisons. Therefore, in this experiment, the participants were asked to make absolute assessments on the quality of only automatic motion effects. Instead, we divided the participants into two groups, the designers group (DG) and the others (OG) for between-group comparisons. DG were expected to give lower scores than OG due to their higher standards.

The questionnaire was the same as that of Experiment I, except for the last question. It was replaced by: Level—"What was the overall level of the motion effects?". All questions were rated as a score between 0 and 10. The point 10 meant that the motion effects were in the same level as the final product designed by the 4D experts. We also had interviews with the participants regarding the quality of our synthesized motion effects after the experiment. The experiment took about 45 min per participant.

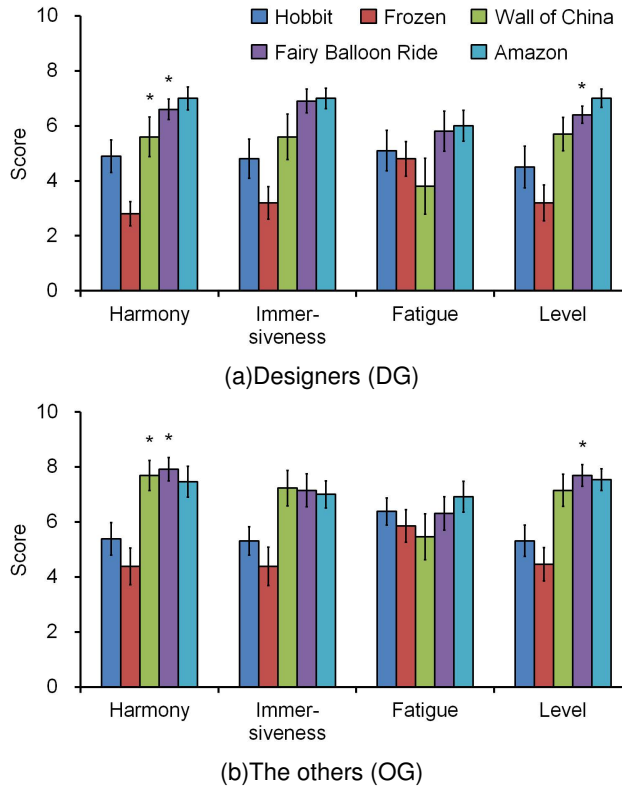


Fig. 5.10 Results of the user study with 4D experts. Error bars represent standard errors. The scale for fatigue is inverted as in Fig. 5.8. Asterisks indicate that the results between DG and OG had statistically significant differences.

Results

The results of the evaluation are shown in Fig. 5.10. Overall, the CF motion effects for the three 4D rides were highly rated; the scores of the four measures ranged from 3.8 to 7.0 with DG and from 5.5 to 8.0 with OG. The lowest scores were for fatigue. The scores for level were 5.7–7.0 with DG and 7.2–7.7 with OG. These scores, especially those of OG, can be considered as very high; people are generally reluctant to give the highest or lowest scores in Likert-scale questions.

The CS motion effects for the two regular 4D films obtained lower yet moderate scores: 2.8–5.1 with DG and 4.4–6.4 with OG. The scores for level were 3.2–4.5 with DG and

4.5–5.3 with OG. Participants commented that the motion effects for Frozen and Hobbit received lower scores than the others because of the absence of class O effects for the motion of the main character. Since less O effects were expected in Hobbit due to its short playback time, its scores were higher than those of Frozen.

The scores between DG and OG were compared using a paired t-test ($\alpha = 0.05$) for each 4D film and each subjective metric. Despite the generally lower scores of DG, significant differences were found only in harmony for Wall of China and Fairy Balloon Ride and in level for Fairy Balloon Ride.

In the participant interviews, all experts agreed that our algorithms would be helpful for motion effects production. They also commented that the level of our synthesized motion effects was greatly higher than their initial expectation. Overall, the evaluation results with 4D experts are promising for the applicability of our synthesis algorithms to actual 4D effects production.

5.4 Discussion

Our work has been carried out independently of Shin et al. [79]. As such, there exist both similarities and dissimilarities between the two system. Most of all, our synthesis algorithms are tailored to the extensive surveys that we executed for a systematic classification of various 4D motion effects in actual use and their design practice in the industry. This approach is expected to improve immersiveness of 4D films most effectively and also the likelihood of our algorithms being adopted by 4D effects designers. In contrast, the framework in [79] is based more on the conventional principles of motion simulation for training purposes, in which the provision of physically exact sensory stimuli has the top priority. This appears to be the most fundamental underlying difference between the two systems.

For example, both systems provide motion effects synthesis algorithms for CF effects, but only our system considers CS and V effects. CF effects have been used in training simulators and 4D attractions for a long time, but the other two are relatively new effects and more frequently used than CF effects in regular 4D films (Fig. 3.3). As for CF effects, the two systems follow the same approach. However, in our system, the computation of

camera velocity and acceleration is done directly from the relative camera motion in the local coordinate frame obtained from two consecutive images. In [79], it is done from the 3D camera position in the world frame estimated by an additional reconstruction step from the relative camera motions. This method resembles VR-based motion simulations where the 3D trajectories of all virtual objects are known, which enables the rendering of sustained low-frequency force such as gravity using tilt coordination. However, the computational load becomes greatly higher, which significantly degrades usability as authoring software that requires iterative design. Our method is much more efficient, but it cannot estimate fictitious forces applied to the camera, sacrificing gravity rendering. This was a strategic decision made also considering that scenes where only gravity effect is dominant are scarce in 4D films and the workspace of commercial motion chairs is quite limited and is not suitable for clear tilt coordination effects, as described earlier in Section 5.1.2. This difference, and also the fact that all our algorithms are integrated into one optimized system while the system in [79] requires several independent software packages to be used in a series, seem to contribute to the substantial difference in synthesis efficiency; for example, our algorithms for CF effects are 50 times faster than [79] (a rough comparison between Section 5.2.2 of our paper and Section 6 of [79]).

In addition, our current synthesis algorithms have the following limitations: 1) Algorithm CF sometimes does not generate sufficiently strong motion commands for the acceleration/deceleration and the heave motion of camera. This stems from the inherent limitation in camera motion estimation that translation estimation is much less accurate than rotation estimation; 2) Algorithm CS offers only one option (linear interpolation) for connecting motion segments. More options (e.g. cubic interpolation) might be necessary for 4D designers' selection to consider the context; 3) Algorithm V also responds to rough human voice, although this can be avoided by careful film segmentation prior to effect design; and 4) The presence of very abrupt camera motion and significant lighting variation may substantially increase errors in optical flow, and such errors can inject perceivable false cues to motion effects, although such extreme occasions are rare in movies.

Synthesis of Object Class Motion Effects

6.1 Overview of Interactive Motion Effects Design

Fig. 6.1 shows an example user interface of interactive motion effects design, the final outcome of this work. A user can design motion effects for a moving object by carrying out the following simple tasks. The user annotates a bounding box for the object of interest then an object tracking algorithm estimates the position of the object in the following frames. A circle, the radius of which is the same as a predefined error threshold, is displayed at the estimated position of the target object. When the center of the target object goes outside of the circle, the user has to stop the tracking, revise the bounding box to enclose the object tightly, and then restart the tracking.

Our interactive motion effects design tool enables even non-experts to quickly design perceptually plausible motion effects; one minute of motion effects can be designed in 15–20 minutes. In contrast, three to eight hours of work by an experienced designer is required to make the same length of motion effects. Although the motion effects designed by our algorithm need to be revised by expert designers for the final release, our approach is expected to greatly improve the productivity of 4D effects authoring.



Fig. 6.1 Example user interface of interactive motion effects design. When the center of the object being tracked moves outside the yellow circle, the designer fixes its bounding box before clicking the ‘Track next frame’ button.

6.2 Rendering algorithms for Object Motion

There are several motion rendering techniques for the visual content recorded in POV shots (C class according to our classification), such as flight simulator [69], driving simulator [70], and 4D attractions [83, 79]. However, there are no prior studies about O class motion effects to our knowledge. For that, we identified two representative rendering strategies—object- and viewer-centered rendering, and then compared their performance by a user study in order to determine which strategy to use for automatic generation of O effects. Details are described in this section.

6.2.1 Object- and Viewer-Centered Rendering

Object-centered rendering generates physically plausible motion effects from the viewpoint of the object of interest. In this case, motion effects appropriate for the typical events are relatively straightforward (see Fig. 6.2c for the coordinate system of 3 degree-of-freedom (DOF) motion chairs): (1) acceleration \rightarrow leaning backward (chair pitch decrease), and (2) deceleration \rightarrow leaning forward (chair pitch increase), (3) left turn \rightarrow tilting left (chair

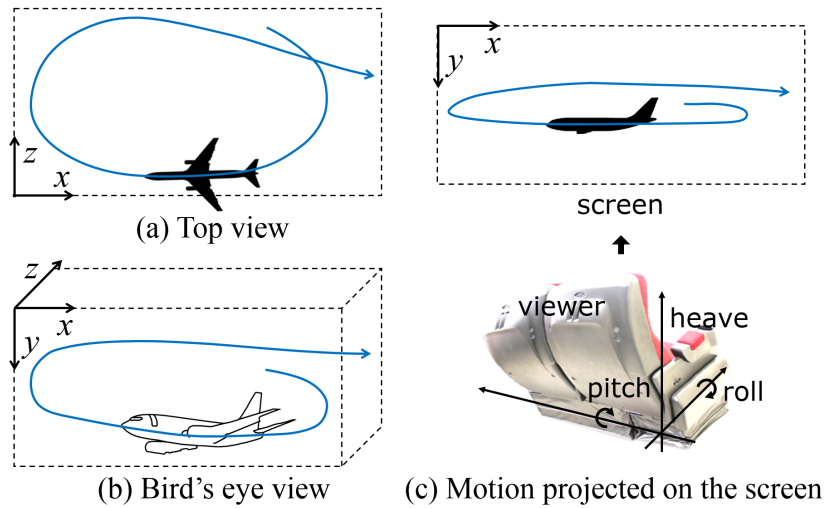


Fig. 6.2 Object- and viewer-centered rendering in an example plane scene.

roll decrease), (4) right turn \rightarrow tilting right (chair roll increase), (5) ascent \rightarrow lifting (chair heave increase), and (6) descent \rightarrow falling (chair heave decrease). (1) and (2) are common strategies to render inertial forces induced by accelerations. (3)–(6) are widely accepted in games and 4D films to mimic active motions of an object¹.

Viewer-centered rendering aligns the chair's motion with that of the viewer's visual attention. This method is based on the assumption that the viewer's visual attention follows the object of interest. Therefore, given motion of the object on the 2D screen, we create motion effects as follows: (1) vertical translation \rightarrow pitch increase/decrease and (2) horizontal translation \rightarrow roll increase/decrease. Pitch is employed instead of heave in (1) because humans also rotate the face vertically to follow a vertically moving object. We used roll to render horizontal motion instead of yaw since yaw is not available in typical motion chairs for 4D films [86]. Viewer-centered rendering is much easier for automation than object-centered rendering since only the 2D position of the object is required.

Fig. 6.2 shows an example scene of a plane that turns right in a circle and then flies

¹The opposite motion effects (e.g., tilting right for a left turn) are used in flight and driving simulators to render centrifugal forces [83].

straight while speeding up. In object-centered rendering, a chair tilts right during the heading change and then leans backward to render inertia induced by the acceleration in the frontal direction. In viewer-centered rendering, a chair tilts right, left, and then right in roll with only a small amount of motion in pitch to follow changes in the gaze direction (Fig. 6.2c).

6.2.2 Comparative User Study

We performed a user study to compare the perceptual quality of motion effects synthesized by object- and viewer-centered rendering.

Methods

Participants were 21 students (11 male and 10 female; 21-30 years old) recruited from the authors' institution. Six participants had limited experiences of watching regular 4D films, three times at most, but the other participants had none. Each participant was paid about 13 USD for this user study.

The motion chair (4DX; CJ 4DPLEX) used in the experiment has 3 DOF for roll ($\pm 4^\circ$), pitch ($\pm 7^\circ$), and heave (± 4 cm). 3D images were projected onto a 94-inch screen by an EB-W16SK polarized 3D projector (Epson corp.). The participants wore passive 3D glasses during the experiment. Sound was played through NS-150 5-Channel home theater speakers (Yamaha corp.).

We used 18 video clips trimmed from various films for the experiment. As shown in Table 6.1, most videos were very short (3.8 s on average) and contained only a few primitive motions. This allows participants to focus on evaluating motion effects for a target object while minimizing the effects of other factors (e.g., mood of a scene) that are not related to motion. These videos include various types of motions and target objects. 36 motion effects for the 18 video clips were designed manually by the first author, which took three days including substantial efforts for quality control.

The experiment consisted of 18 sessions, one for each video clip. In each session, the participants watched a video clip six times while object- and viewer-centered motion ef-

Table 6.1 List of videos used in the comparative user study. If object- and viewer-centered rendering were significantly different ($\alpha = 0.05$), the serial number of the video is marked with a bold text. Blue and red number represent object- and viewer-centered rendering was better, respectively.

#	description of motion	object	duration
01	big left turn with descent and then ascent	iron man	3.2 s
02	big right turn with descent	plane	3.2 s
03	short left turn and then rotations in roll	car	2.3 s
04	short left turn	car	2.1 s
05	right turn and then u-turn	plane	4.6 s
06	right turn, acceleration, and then left turn	helicopter	3.5 s
07	left, right, and then big left turn	dragon	5.8 s
08	left, right, and then left turn with ascent	bird	10.6 s
09	left, right, left turn, and then descent	plane	5.0 s
10	right, left, right, and then left turn	dragon	4.3 s
11	sudden acceleration with ascent	iron man	1.8 s
12	acceleration	plane	2.7 s
13	acceleration with left turn	iron man	1.3 s
14	acceleration	dragon	2.9 s
15	sudden acceleration	iron man	2.0 s
16	four accelerations and then right turn	plane	8.4 s
17	descent, sudden rotation in yaw (drift), and then sudden stop	car	2.2 s
18	right turn, acceleration, deceleration, and then sudden right turn	iron man	2.0 s

fects were presented alternately. The order of the motion effects was randomized for each participant. After each session, the participant took at least 3-min rest to prevent motion sickness and fatigue. The experiment took 90–110 min for each participant.

After watching each video clip six times with object- and viewer-centered effects, the participants answered a questionnaire that included the following four questions:

Harmony Did the motion effects matched to the motion of the main character?

Fun Did the motion effects make the film fun?

Immersiveness Did the motion effects help you be immersed in the film?

Comfort Did the motion effects feel comfortable?

Participants were asked to rate for all questions on a continuous scale by selecting a position on a 14-cm horizontal line. The two ends of the horizontal lines were labeled with symmetric positive and negative ratings, e.g., for harmony, “very inharmonious” at the left end and “very harmonious” at the right end.

Results

We performed paired t -tests to compare object- and viewer-centered rendering. Among the 18 video clips, we observed statistically significant differences in 8 video clips, as illustrated in Fig. 6.3. Viewer-centered rendering has better scores than object-centered rendering in 6 video clips out of the 8 cases. This result suggests that viewer-centered rendering generates better or similar motion effects in terms of perceptual quality compared to object-centered rendering in the majority of scenarios.

Object-centered rendering was perceptually preferred only in two videos, #03 and #17. The one thing the two videos have in common but all the others do not is that they involve significant orientation changes of target objects (in roll and yaw) rather than position changes. Viewer-centered rendering has limitation in handling this kind of scenario since it only considers 2D position changes in the image plane. On the other hand, viewer-centered rendering was better in most of general situations. This is probably because human feels more natural to understand scene from the viewpoint from the third person unless the scene is a POV shot.

Another great advantage of viewer-centered rendering is that implementation is much easier since there are many visual tracking algorithms we can adopt for 2D motion estimation. To implement object-centered rendering, however, 6 DOF motion in the world coordinate should be estimated from image sequences. Obviously, this is substantially more challenging although there are several available algorithms for pose tracking [66, 45], but their computational efficiency and robustness to challenging conditions are still not satisfactory.

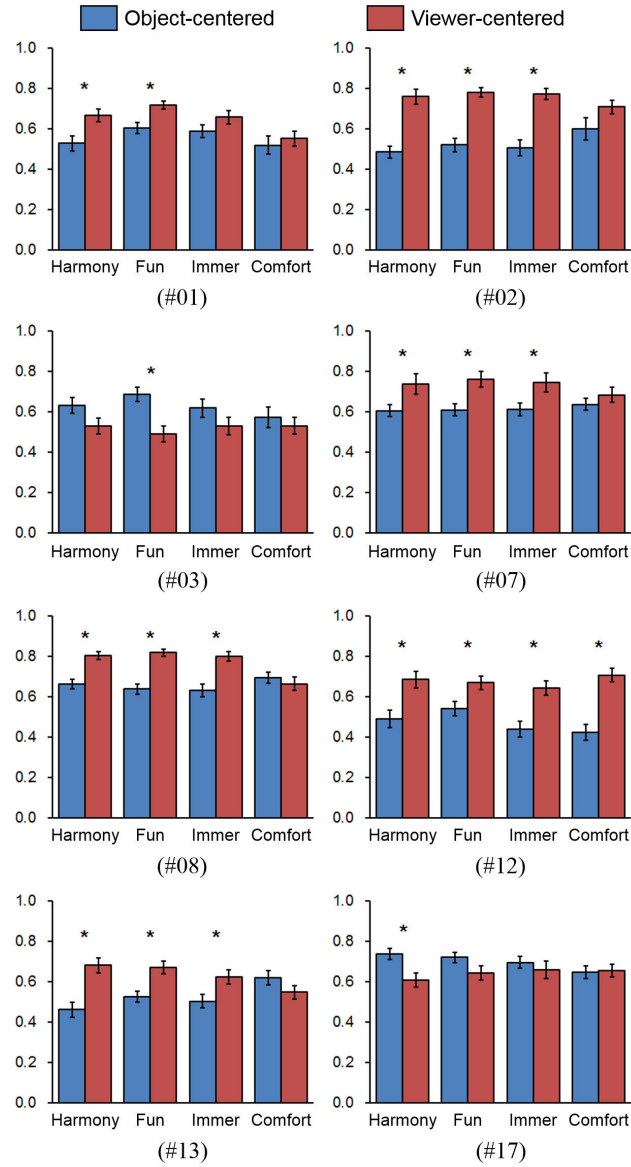


Fig. 6.3: Results of comparative user study between object- and viewer-centered rendering. Among total 18 videos, only 8 statistically significant ($\alpha = 0.05$) results are illustrated. Results of remaining 10 videos are presented in Fig. 6.4. Error bars represent standard errors. Asterisks indicate that the two rendering methods were statistically significant.

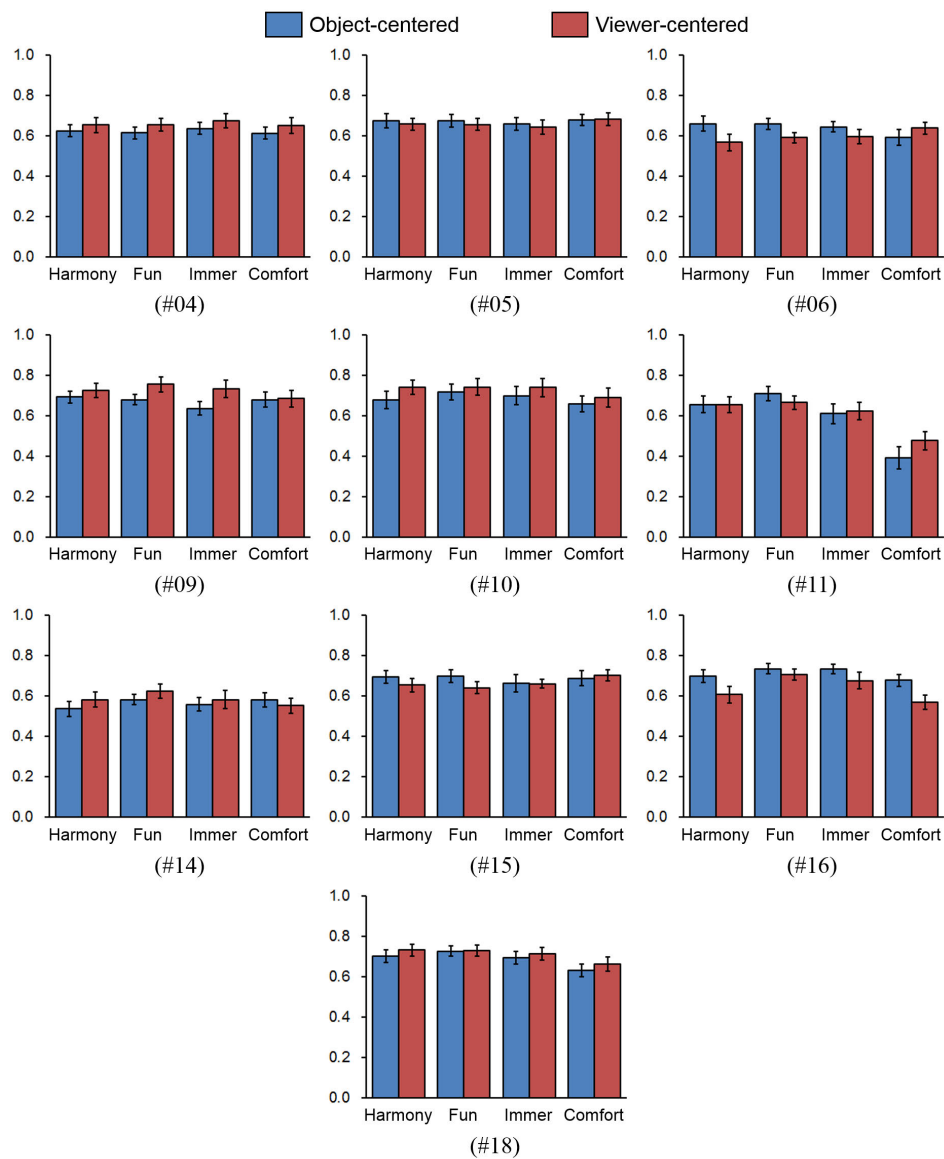


Fig. 6.4: Results of comparative user study between object- and viewer-centered rendering. Among total 18 videos, 10 statistically not significant results are illustrated. Error bars represent standard errors.

6.3 Implementation of Viewer-Centered Rendering

Motivated by the results of the comparative user study, we implemented an O effects synthesis algorithm based on the viewer-centered rendering. For this purpose, our algorithm estimates 2D positions of the object of interest using an object tracking algorithm. To show the effectiveness of trajectory estimation by automatic tracking algorithms, we also consider a cubic spline, which smoothly interpolates ground-truth target positions annotated by humans for sparsely sampled frames. Mapping between the estimated positions of an object and the motion commands for a 4D chair is performed by a standard washout filter algorithm [60].

The following subsections present the overall procedure of viewer-centered rendering based on our algorithm in detail. We first discuss how we implement the motion effects based on the washout filter, and then describe the procedure to estimate 2D positions of an object through interactive tracking.

6.3.1 Washout Filter Algorithm

Viewer-centered rendering is implemented using a classical washout filter, the de facto standard in motion simulators [60]. The motion chair has a limited workspace, so it has to return to its origin after generating an effect in order to be ready for the next effect. To handle this problem, the washout filter is designed to be essentially a set of high-pass filters that renders high-frequency onset cues, removing the low-frequency energy that pushes the motion chair to its workspace limit.

Fig. 6.5 illustrates our washout filter algorithm in detail. First, the 2D positions of a target object obtained by object tracking or cubic spline are converted to smoothed linear velocities using Kalman filter. To implement viewer-centered rendering, the horizontal and vertical velocities of the object (v_x and v_y) are converted to the roll and pitch commands of the motion chair (p_r and p_p), respectively. Scale factors c_x and c_y work as gains for motion effects, where we used 0.05 and 0.07 as default values, respectively. c_y is larger than c_x to utilize the larger limit of pitch of our motion chair (± 4 and ± 7 degrees for roll and pitch,

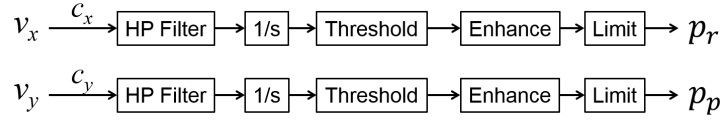


Fig. 6.5 Washout filter algorithm for viewer-centered rendering.

respectively). A first-order Butterworth high-pass filter with a cutoff frequency of 1.0 Hz is used. Soft thresholding is applied to remove small noises during the stationary state. We further enhance the motion commands to obtain more dynamic and fun motion effects. A motion command m is amplified as

$$\hat{m} = \text{sgn}(m) \alpha \left(\frac{\|m\|}{\alpha} \right)^\beta, \quad (0 < \beta \leq 1) \quad (6.1)$$

where \hat{m} is an enhanced motion command and α is the maximum angle of the motion. Decreasing β makes motion effects more dynamic. According to our experience, $\beta \in [0.7, 0.9]$ is reasonable for most 4D films. We limit the motion commands to the maximum displacement and velocity of the motion chair for safety.

6.3.2 Selection of Object Tracking Algorithm

Object tracking is a natural choice to obtain the 2D positions of a target object from image sequences. We selected four recent tracking algorithms and evaluated their performances to identify the best one. The four candidates include SCM [94], Struck [23], CN [15], and MEEM [92]. SCM and Struct are the top two tracking algorithms out of 29 methods in online tracking benchmark [87]. CN and MEEM are newer techniques that show excellent performance for the same benchmark.

We collected 17 test sequences (48–420 frames) whose average length is 192 frames, and all image frames were rescaled to 640 pixel wide. The sequences were extracted from 10 different films, and most of them are action and adventure movies appropriate for 4D effects. These test sequences are much more challenging than the ordinary videos used for tracking performance evaluation. Our videos often contain large motions, scale changes,

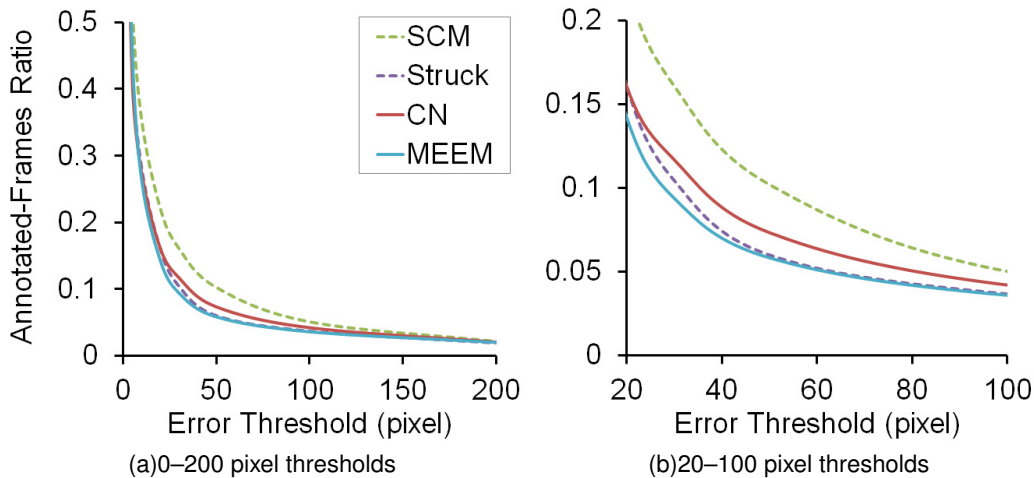


Fig. 6.6 The ratio of annotated frames with respect to various tracking error thresholds. The smaller is the better. MEEM was better than the other compared algorithms when evaluated using center location errors. (a) and (b) are the same results, but (b) shows the results in more detail for error thresholds 20–100 pixel.

pose variations, illumination changes, occlusions, motion blurs, etc. All the tested tracking algorithms were not successful in most of our sequences and often lost target objects completely within the first 10 frames. MEEM was slightly better than other algorithms and reasonably tracked the targets in two test sequences.

The disappointing performance of existing tracking algorithms leads to the need of user interventions in our algorithm. Whenever a tracking algorithm fails, we reinitialize the target and restart to track from the failed frame. The four tracking algorithms were evaluated under this use scenario, where we counted the number of annotations (reinitializations) required to successfully track all frames in each sequence. Success or failure was determined by the distance between the center positions of the ground-truth and the estimated object bounding boxes.

Fig. 6.6 illustrates the annotated frame ratios of all algorithms with respect to various center position error thresholds from 0 to 100 pixels. MEEM showed the best performance consistently for all the thresholds while SCM was the worst with the same measure; SCM requires 40–75% more annotations than MEEM in the tested threshold range. This evalua-

tion supports the use of MEEM as the default tracking algorithm for target position estimation.

6.3.3 Motion Effects Design Using Object Tracking

As shown in Fig. 6.6, there is a trade-off between tracking error threshold and the number of annotations (i.e., the amount of human efforts). If two motion effects generated based on two tracking results with different numbers of annotations have similar perceptual quality, it is obvious that the motion effects designed with less annotations are preferred. We carried out user experiments to understand the impact of the accuracies of tracking algorithms and the variations of the error thresholds to the perceptual quality of motion effects, and obtained a reasonable guideline from the experiments to generate effective 4D motion effects. The procedures and results of these user studies are as follows.

Methods

For this user study, we recruited 20 students (13 males and 7 females; 19-31 years old) from the institution where this research was conducted. Nine participants had limited experience in watching regular 4D films, three times at most, while the others had no experiences. Each participant was paid about 13 USD after this user study.

We used five video clips in this experiment, which are extracted from the following movies: *The Avengers* (video duration 8.4 s), *Bolt* (4.1 s), *How to Train Your Dragon 1 and 2* (13.9 s and 7.0 s), and *Tangled* (10.3 s). These videos are generally longer and involve more complex motions than the 18 video clips used in the previous comparative user study. Eight different versions of motion effects for each video clip were synthesized by our viewer-centered rendering algorithm using the combinations of four tracking error thresholds (20, 50, 100, and 200 pixel) and two tracking algorithms (SCM and MEEM).

The experiment consisted of five sessions, one for each video. In each session, one video clip was presented eight times with different sets of motion effects. The order of the motion effect sets was randomized for each participant. The questionnaire and rating method were the same as those in the user study reported in the previous section. After each session, the

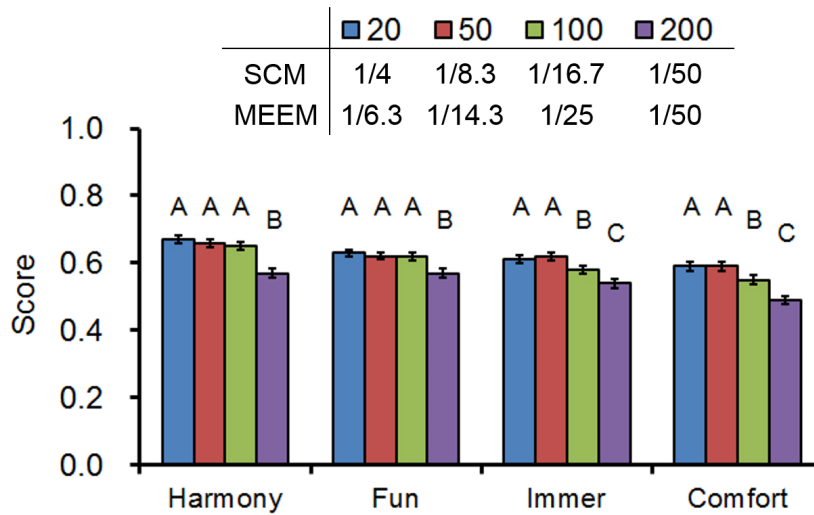


Fig. 6.7 Results of the user experiment for the variations of tracking error thresholds (in pixel). Error bars represent standard errors. The set of motion effects marked with the same alphabets indicates that they did not show statistically significant differences by the SNK tests. The table shows an average frame annotation ratio for each experimental condition.

participants took at least 3-min rest. The experiment took 70–80 minutes per participant.

Results

Experimental results are shown in Fig. 6.7. As expected, more accurate tracking led to better perceptual quality of motion effects. We performed three-way ANOVA on each question using tracking error threshold, tracking algorithm, and video as the independent variables. Error threshold and video were statistically significant ($\alpha = 0.05$) while tracking algorithm was not for all the subjective metrics. The Student-Newman-Keuls (SNK) test showed that the motion effects with 20 and 50 pixel error thresholds were not significantly different in all questions. Motion effects with 100 pixel error threshold were significantly worse than motion effects with 20 and 50 pixel error threshold in immersiveness and comfort. Motion effects with 200 pixel error threshold were significantly worse in all the subjective metrics compared to those of other error thresholds. Although there were some interaction effects, we could not find further generalizable conclusions from them. All interaction effects were

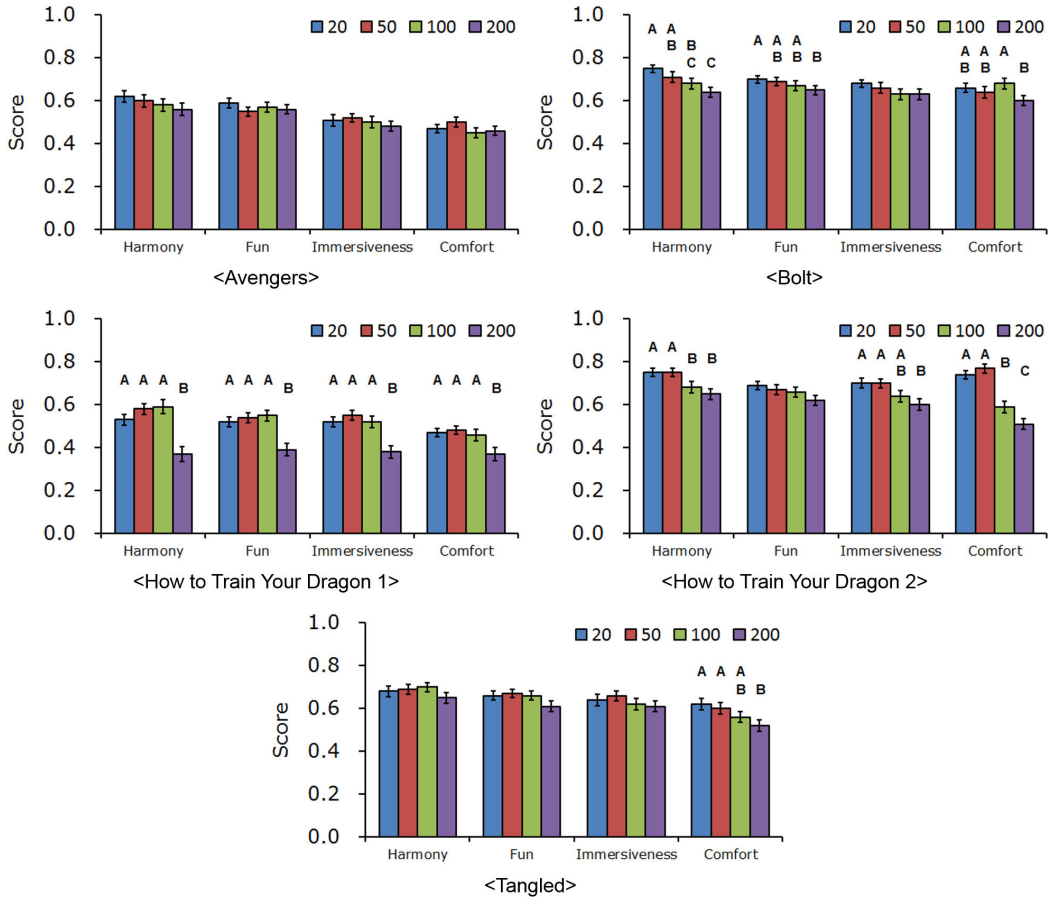


Fig. 6.8 The results of individual videos for different tracking error thresholds. Error bars represent standard errors. The set of motion effects marked with the same alphabets indicate that they did not show statistically significant difference by SNK test.

related to the specific choice of the video clip.

Although the type of tracking algorithm was not a statistically significant factor, MEEM was superior than SCM in terms of annotation effort as shown in the table above Fig. 6.7. For example, 70% more annotations are necessary for SCM compared to MEEM with respect to 50 pixel error threshold.

Fig. 6.9 shows examples of the motion effects used in the experiment. Motion effects

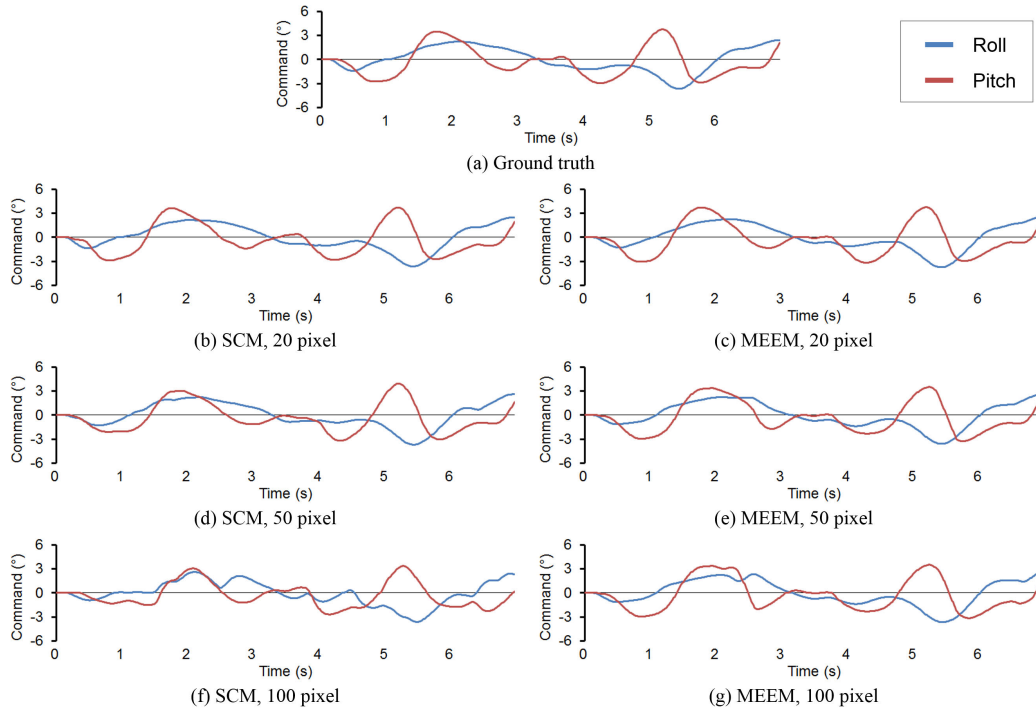


Fig. 6.9: Examples of the motion effects generated by SCM and MEEM with different tracking error thresholds for *How to Train Your Dragon 2*. (a) shows motion effects synthesized based on the ground truth positions of the object.

generated with 50 pixel error thresholds look very similar those generated with the ground truth. From the 100 pixel error thresholds, differences from the ground truth motion effects are obvious, especially in SCM. Fig. 6.10 presents the correlation coefficients between the motion effects generated by the ground-truths and the motion effects designed by tracking algorithms with different error thresholds; the results clearly show the tendency found in Fig. 6.9.

In conclusion, our algorithm is robust to the type of tracking algorithm and tracking error. We recommend to use a tracking error threshold of 50 pixel to ensure motion effects of good perceptual quality. In this case, it seems that the type of tracking algorithm does not significantly affect the quality of motion effects. However, more accurate tracking algorithm is preferred to reduce the time required to design motion effects. Tracking with

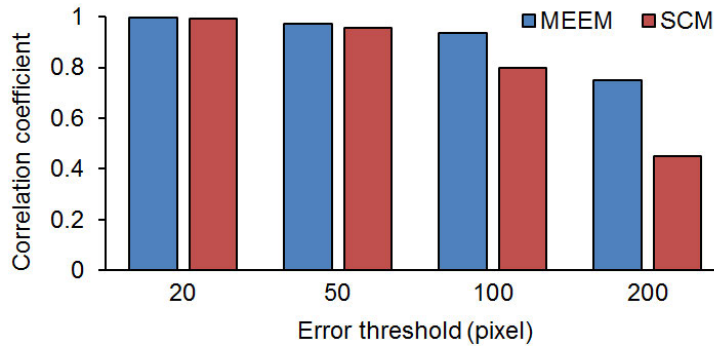


Fig. 6.10 Correlation coefficient between the motion effects generated by the ground truth and the motion effects generated by tracking with different error thresholds. Average correlation coefficient for five video clips were taken.

up to 100 pixel error threshold is also not bad, but robust tracking algorithms, such as MEEM, should be used.

6.3.4 Motion Effects Design Using Spline

According to the previous experiment, motion effects should be designed with 50 pixel (or less) error threshold to obtain good results. It means that one annotation for every 17.2 frames on average is necessary (see Fig. 6.6). Since it is not a small number of annotations, motion effects can also be designed by a spline that interpolates annotated positions smoothly. If the motion effects generated by spline interpolation have a similar level of perceptual quality in comparison to motion effects designed by object tracking, the former is better than the latter in terms of the simplicity of implementation. We performed a user experiment to evaluate the quality of motion effects designed based on the natural cubic spline.

Methods

The participants were 21 students (11 males and 9 females; 18-30 years old) recruited from the author's institution. Eleven participants had experiences of watching regular 4D film, five times at most, and the other had not. Each participant was paid about 9 USD after the

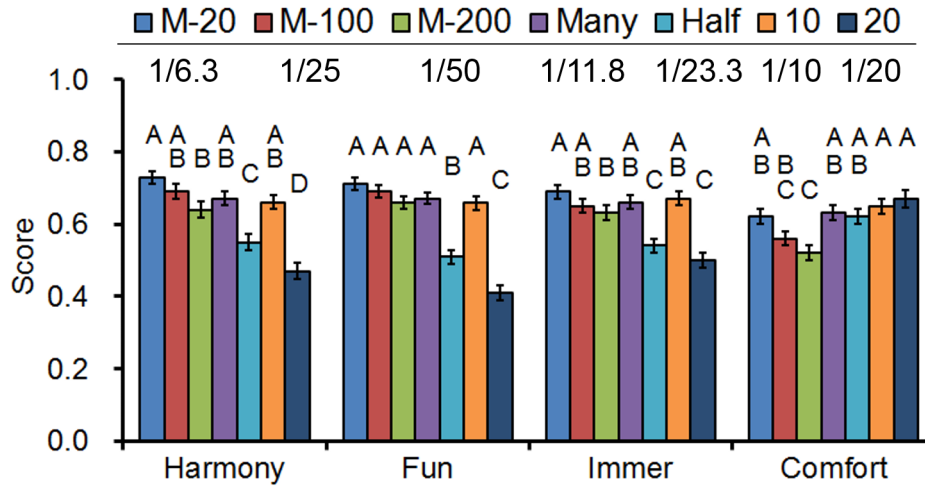


Fig. 6.11 Results of the user experiment that compared the motion effects designed by object tracking and natural cubic spline. Error bars denote standard errors. The set of motion effects marked with the same alphabets represent that they did not show statistically significant differences by the SNK tests. The table shows average frame annotation ratios for each experimental condition.

experiment.

The same video clips used in the previous experiment were used in this experiment. The motion effects were generated by viewer-centered rendering based on the 2D positions estimated by the natural cubic splines that smoothly connected the centers of annotated bounding boxes. We tested the following four different bounding box annotation strategies: (1) Many–The designer determines the frames to be annotated for the best motion effects, (2) Half–It is same with Many except that only a half of bounding boxes are annotated compared to Many, (3) 10–The designer annotates every 10-th frames, and (4) 20–The designer annotates every 20-th frames. The motion effects generated by MEEM tracker with 20, 100, and 200 pixel error thresholds were also included in the experiment. This experiment took 50-60 min per participant.

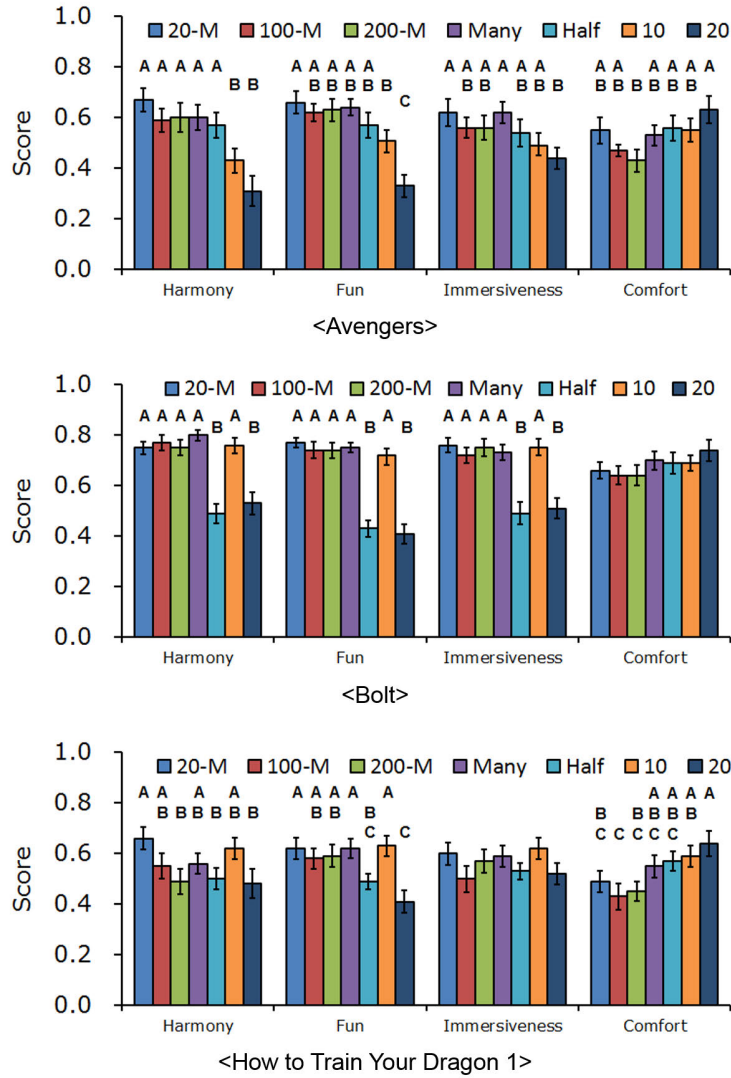


Fig. 6.12 The results of individual videos (The first three videos). Error bars denote standard errors. The set of motion effects marked with the same alphabets represent that they did not show statistically significant difference by SNK test.

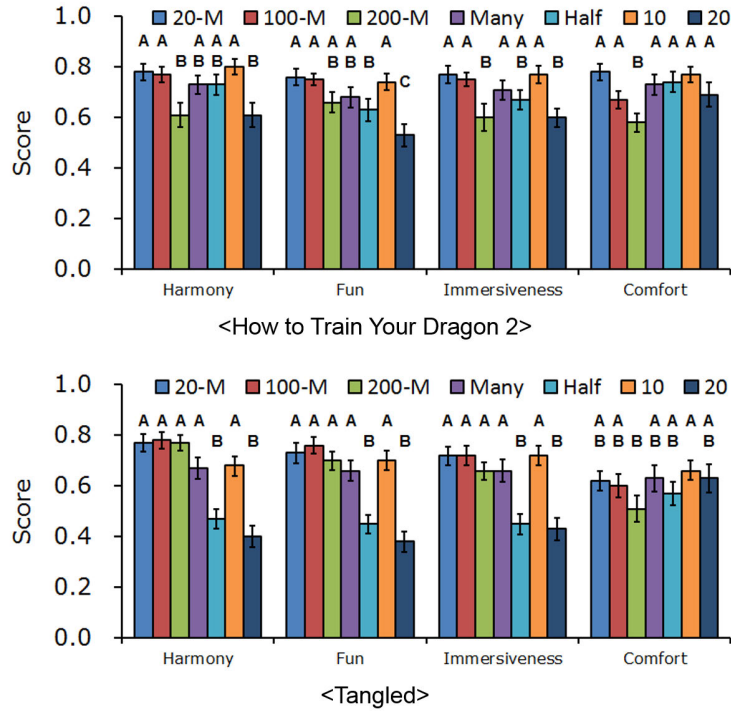


Fig. 6.13 The results of individual videos (The remaining two videos). The set of motion effects marked with the same alphabets represent that they did not show statistically significant difference by SNK test.

Results

Fig. 6.11 shows the results of the experiment. We performed two-way ANOVA on each question using design method and video as the independent variables. Design method, video, and their interaction were statistically significant ($\alpha = 0.05$) for all the subjective metrics. According to the SNK test, motion effects designed by spline interpolation with a sufficient number of annotations showed comparable perceptual quality to the motion effects designed by the tracking algorithm with less than 100 pixel error threshold. The quality of motion effects was significantly degraded when the motion effects were generated by spline interpolation with only a small number of annotations, Half and 20.

Even though object tracking and natural cubic spline did not induce significant differ-

ences in perceptual quality of motion effects, object tracking has greater advantage than natural cubic spline in terms of annotation effort. Motion effects designed by MEEM-100, Many, and 10 have comparable perceptual quality, but Many and 10 require 2–3 times more annotations than MEEM-100 as shown in Fig. 6.11.

In some scenes, the motion of the object of interest is extremely fast and difficult to track using existing tracking algorithms. In this case, using natural cubic spline can be a better choice. A designer should annotate the position of the object for more than one frame for every 10 frames to design motion effects of good quality.

6.4 User Experiments

Lastly, we performed two more user experiments to assess the perceptual quality of our motion effects design algorithm, one with general users and the other with 4D experts.

6.4.1 Experiment I: General Users

The goal of Experiment I was to compare the benefits that general users perceive from different sets of motion effects: those randomly generated, synthesized by our algorithms, and manually designed by 4D experts.

Methods

The participants were 22 students (13 male and 9 female; 18–27 years old) recruited from the authors' institution. None of them reported prior involvement in 4D effects production. Thirteen participants had limited experience of watching 4D films, five times at most, while the others had none. Each participant was paid about 9 USD after the experiment.

Three video clips, which have a much longer playback time than the video clips used in the previous experiments, from regular 4D films were used in this experiment: *The Avengers* (3 min 5 s), *How to Train Your Dragon 2* (2 min 24 s), and *Planes: Fire & Rescue* (1 min 14 s).

Four sets of motion effects were used in the experiment. RE was randomly generated effects using Perlin noise. OE was O class motion effects designed by viewer-centered

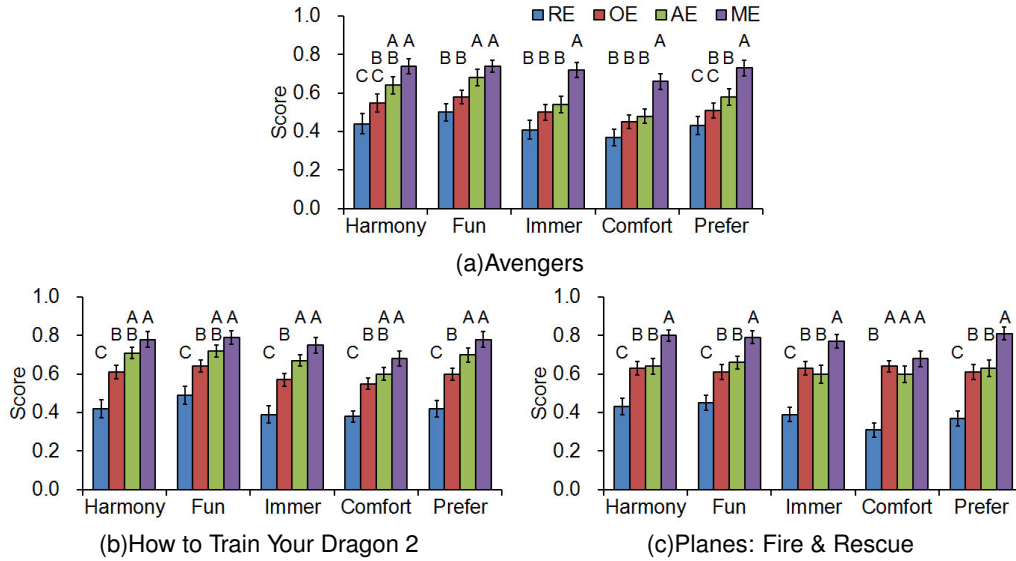


Fig. 6.14: Results of Experiment I with general users. Error bars represent standard errors. Alphabets represent SNK grouping.

rendering with MEEM of 50 pixel error threshold. AE was an addition of all O, C (camera), and V (impulse and vibration) class effects. ME was manually designed by an experienced 4D designer. It is noted that all classes of motion effects were included in ME according to the designer's discretion.

The experiment consisted of three sessions, one for each 4D film. In each session, one 4D film was presented four times with different sets of motion effects. The order of the 4D films and the motion effects sets was randomized for each participant. After each session, the participant took at least 5-min rest. The experiment took 50–60 min per each participant.

The questionnaire was the same as that of the previous experiments, but one more question was added: Preference—"Did you like the motion effects provided with the film?".

Results

Experimental results are shown in Fig. 6.14. We performed one-way ANOVA on each question using motion effects set as the independent variable for each 4D film. Motion effects set was statistically significant ($\alpha = 0.05$) for all the subjective metrics and all the

4D films. The SNK tests were performed for post-hoc multiple comparisons.

ME showed the best scores in all the metrics and the conditions. AE was the second and received comparable scores to ME especially in How to Train Your Dragon 2 (Fig. 6.14b). Between AE and OE, AE obtained higher scores due to the inclusion of C and V class of motion effects. Statistically significant differences were observed only in fun for The Avengers, and immersiveness and preference for How to Train Your Dragon 2. RE received the lowest scores in all cases.

In conclusion, in terms of perceptual quality to general users, the motion effects designed by our algorithms outperformed the random effects, and were even comparable to the manually designed motion effects in some cases. Even though all classes of motion effects were combined together in ME, the results are sufficient to show that our interactive motion effects design is able to synthesize perceptually plausible O effects, since O effects were much more frequent than the other classes in the videos used in the experiment.

6.4.2 Experiment II: 4D Experts

Experiment II was to assess the quality of motion effects designed by our algorithms with the careful eyes of 4D experts.

Methods

Six 4D experts (4 male and 2 female; 28-32 years old) who work for CJ 4DPLEX participated voluntarily in this evaluation. Four of them were 4D effects designers, and two of them were software and hardware engineers. The experts had more than two years of work experience in their current position. All of them had ample experience in making or evaluating 4D effects.

The motion effects and films were the same as those used in Experiment I. Since our expert participants could distinguish details in motion effects, they were very likely to notice whether motion effects were manually designed or automatically synthesized. This can instill a substantial bias to blind comparisons. Therefore, in this experiment, the participants were asked to make absolute assessments on the quality of only the motion effects designed

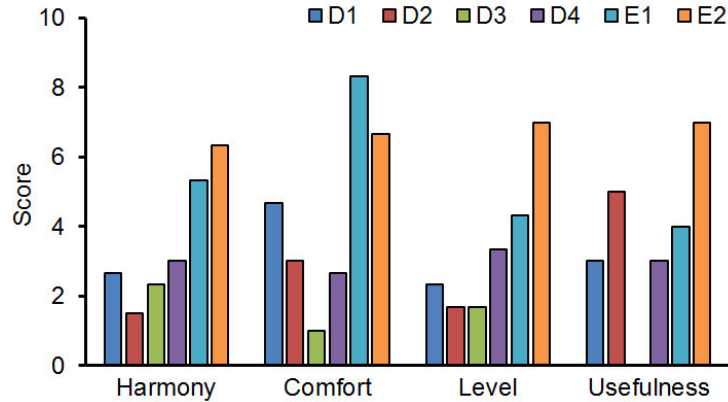


Fig. 6.15 Results of the user study with 4D experts. The scores of harmony, comfort, and level for three video clips were averaged. D1, D2, D3, and D4 are the results of designers and E1 and E2 are the results of developers.

by our algorithms.

The questionnaire consisted of three questions: Harmony, Comfort, and Level. The questions about harmony and comfort were the same to that of Experiment I. The question for level was “What was the overall level of the motion effects?”. After evaluating all three films, the participants were asked to answer one more question: Usefulness—“Do you think our current algorithm is useful for designing 4D effects?”. All questions were rated using a score between 0 and 10. The point 10 meant that the motion effects were in the same level as the final product designed by 4D experts. We also had interviews with the experts regarding the quality of motion effects after the experiment. The experiment took about 30 min per participant.

Results

The results of the evaluation are shown in Fig. 6.15. Overall, scores rated by designers were low (1.0–4.7) while scores rated by engineers were good (4.3–8.3). The scores for usefulness were 0–5 with designers and 4–7 with engineers. It was expected since expert designers have much higher standards than engineers and general users. Moreover, expert designers are inherently unfavorable to automatic algorithms because they consider their

work as highly artistic tasks. Even though the evaluation by the expert designers was not good enough, it shows promising potential of our algorithms, considering very high standards of expert designers.

There is one more possible reason for the low scores of expert designers. The in-house authoring tool they used does not support easy modification of the existing motion effects; expert designers have to overwrite the existing motion effects to modify them. If our algorithms are incorporated with convenient editing functions of an authoring tool, we may receive much more positive response from the designers.

The following is a summary of the weaknesses described by the experts after the experiment: (1) Only roll and pitch motions are used. Heave is essential for better motion effects, e.g. to express a sudden fall of the target object. (2) More delicate control of the strength of motion effects is required. For example, weaker motion effects are favorable for less important target objects. (3) There were no motion effects for the spin of objects. (4) Motion effects were too discrete for each event. Connecting motion effects more smoothly and continuously will lead to motion effects of better quality.

(1) and (3) are inherent limitations of viewer-centered rendering that considers only 2D positions of the target object. Object-centered rendering should be employed to handle these limitations. We can solve (2) by allowing a designer to annotate the relative importance of the object in Fig. 6.1. (4) can be alleviated by further smoothing or global optimization process. As future work, we plan to improve the limitations of our algorithm.

Chapter 7

Conclusions

In this paper, we have presented synthesis algorithms that generate motion and vibration effects for 4D film and mobile devices. After categorizing motion and vibration effects into six classes based on surveys, synthesis algorithms for five classes of motion and vibration effects are described.

Our Algorithm V extracts two auditory perceptual metrics, loudness and roughness, from audio signals and then converts them to two vibrotactile perceptual metrics, intensity and roughness. Vibrotactile stimuli that deliver the specified perceptual properties are synthesized by superimposing two sinusoidal vibrations with different frequencies. Our Algorithm CF and CS estimate camera motion and then transform it to vestibular feedback using a washout filter for dynamic POV shots, or to velocity feedback using motion segmentation and scaling for slowly changing scenes. Various motion effect examples made by our algorithms are provided using plots. Our Algorithm O is based on viewer-centered rendering that matches the motion of the chair to the movement of visual attention of a viewer. We performed extensive user studies to find the optimal parameters for implementation while considering both perceptual quality and algorithmic simplicity.

Using our algorithms, the draft of motion effects can be designed quickly (at least 10 times faster than the current manual authoring) even by non-experts. All user experiments have indicated that our algorithms are capable of synthesizing compelling motion effects in

terms of perceptual quality. Even though evaluations on our final results for regular 4D film by expert designers were not satisfactory, it shows promising potential of our algorithms while considering very high standards of the expert designers.

요약문

4D 영화를 위한 모션 및 진동 효과의 자동 생성

4D는 영상에 의자의 모션, 진동, 바람, 물 등 다양한 물리적 효과를 함께 입혀 사용자의 몰입감을 더욱 높여주는 기술을 말한다. 최근 4D 영화 전문 상영관이 늘어나고 모바일 기기, 게임기, 홈시어터 등 개인용 기기로 까지 4D 기술이 퍼지고 있으나 4D 효과 제작은 여전히 전문가의 수작업을 통해서만 이루어지고 있다. 우리는 이러한 문제를 개선하기 위해 영상과 소리를 분석하여 이를 의자의 모션 효과 또는 진동 효과로 자동으로 변환해주는 기술을 개발하였다.

먼저 우리는 4D 효과와 영화에 대한 심도있는 사전 조사를 통해 다양한 4D 효과를 총 6개의 종류로 분류하였고, 이 중 5 종류의 4D 효과에 대한 자동 생성 알고리즘을 개발하였다. 첫번째 분류는 폭발, 충돌 등과 같은 특수 효과에 따라 발생하는 모션과 진동 효과이다. 우리의 알고리즘은 소리를 분석하여 이러한 효과를 생성해 낸다. 소리를 분석하여 자동으로 진동 등의 특수 효과를 생성하는 기존 기술은 대부분 저역 통과 필터와 같은 신호 수준의 단순한 방법을 사용한 것이 대부분이었다. 반면 우리가 제안하는 방법은 신호적 특성을 거칠기, 세기 등과 같은 인지적 특성으로 변환한 후 인지적인 특성만을 고려하여 소리와 진동 또는 소리와 모션 사이의 변환을 수행한다. 인지적인 특성은 그 자체로 사람이 쉽게 이해할 수 있는 의미를 가지고 있으므로 이 방법을 사용하면 더 쉽고 직관적으로 변환 모델을 설계할 수 있으며 모델의 결과 또한 설계 단계에서 부터 쉽게 예측이 가능한 장점이 있다. 두번째 분류는 영상에 나타난 빠른 카메라 움직임에 대해 빠르고 역동적인 모션 효과를 제공하는 것이다.

세번째 분류는 느리고 잔잔한 카메라 움직임에 대해 연속적이고 부드러운 모션 효과를 제공하여 몰입감을 높여주는 것이다. 이 두 가지 분류의 효과를 위해 본 알고리즘은 먼저 컴퓨터 비전 기술을 이용해 영상에서 카메라의 움직임을 복원해 낸다. 그 후 복원해 낸 카메라 모션을 두 가지 서로 다른 방법의 알고리즘을 사용해 적절한 모션 효과로 변환한다. 네번째와 다섯번째 분류는 화면상 3인칭 시점에서 움직이는 물체의 동작에 어울리는 모션 효과를 제공하는 것이다. 우리는 이를 위해 2D 화면상에서 물체의 움직임만을 바탕으로 동작하며 시청자의 예상되는 시선 이동 방향을 따라 의자를 움직여주는 시청자 중심 렌더링 방식을 고안하였다. 세부적인 알고리즘과 설정 사항 선택을 위해 사용자 평가 실험을 여러번 수행하여 최적의 알고리즘을 구현하였다.

본 논문의 방식으로 모션 및 진동 효과를 생성할 경우 기존의 수동 제작 방식에 비해 최소 10배 이상 빠른 속도로 제작이 가능하다. 다양한 방법의 알고리즘 성능 평가와 사용자 선호도 평가 결과 우리가 제안하는 알고리즘이 실제 전문가가 직접 제작한 효과와 거의 비슷한 수준의 우수한 4D 효과를 자동으로 생성해 내고 있음을 확인할 수 있었다.

REFERENCES

- [1] 4DX. <http://www.cj4dx.com/>.
- [2] T. Ahmaniemi, J. Marila, and V. Lantz. Design of dynamic vibrotactile textures. *IEEE Transactions on Haptics*, 3(4):245–256, 2010.
- [3] D. S. Alles. Information transmission by phantom sensations. *IEEE Transactions on Man-Machine Systems*, 11(1):85–91, 1970.
- [4] S. Baker, D. Scharstein, J. Lewis, S. Roth, M. J. Black, and R. Szeliski. A database and evaluation methodology for optical flow. *International Journal of Computer Vision*, 92(1):1–31, 2011.
- [5] S. Bensmaia, M. Hollins, and J. Yau. Vibrotactile intensity and frequency information in the pacinian system: A psychophysical model. *Perception & Psychophysics*, 67(5):828–841, 2005.
- [6] F. Biocca and B. Delaney. Immersive virtual reality technology. In *Communication in the Age of Virtual Reality*, chapter 4, pages 57–126. Routledge, 1995.
- [7] D. M. Birnbaum and M. M. Wanderley. A systematic approach to musical vibrotactile feedback. In *Proc. ICMC*, volume 2, pages 397–404, 2007.

-
- [8] L. M. Brown, S. A. Brewster, and H. C. Purchase. A first investigation into the effectiveness of tactons. In *Proc. WHC*, pages 167–176. IEEE, 2005.
- [9] C. Chafe. Tactile audio feedback. In *Proc. ICMC*, pages 76–79, 1993.
- [10] A. Chang and C. O’Sullivan. Audio-haptic feedback in mobile phones. In *Ext. Abstracts CHI*, pages 1264–1267. ACM Press, 2005.
- [11] D. Chi, D. Cho, S. Oh, K. Jun, Y. You, H. Lee, and M. Sung. Sound-specific vibration interface using digital signal processing. In *Proc. CSSE*, pages 114–117, 2008.
- [12] S. Choi and K. J. Kuchenbecker. Vibrotactiledisplay: Perception, technology, and applications. *Proceedings of the IEEE*, 101(9):2093–2104, 2013.
- [13] CJ 4DPLEX. Absolute cinema experience, 4DX. Brochure.
- [14] D-Box. <http://www.d-box.com/>.
- [15] M. Danelljan, F. S. Khan, M. Felsberg, and J. van de Weijer. Adaptive color attributes for real-time visual tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1090–1097, 2014.
- [16] F. Danieau, J. Fleureau, P. Guillotel, N. Mollet, M. Christie, and A. Lecuyer. Hapseat: Producing motion sensation with multiple force-feedback devices embedded in a seat. In *Proceedings of ACM VRST*, pages 69–76, 2012.
- [17] F. Danieau, J. Fleureau, P. Guillotel, N. Mollet, M. Christie, and A. Lecuyer. Toward haptic cinematography: Enhancing movie experience with haptic effects based on cinematographic camera motions. *IEEE MultiMedia*, 21(2):11–21, 2013.

-
- [18] F. Danieau, A. Lecuyer, P. Guillotel, J. Fleureau, N. Mollet, and M. Christie. Enhancing audiovisual experience with haptic feedback: A survey on hav. *IEEE Transactions on Haptics*, 6(2):193–205, 2013.
- [19] G. A. Gescheider. *Psychophysics: The Fundamentals*. Lawrence Erlbaum, 3rd edition, 1997.
- [20] B. R. Glasberg and B. C. J. Moore. A model of loudness applicable to time-varying sounds. *Journal of Audio Engineering Society*, 50(5):331–342, 2002.
- [21] F. E. Guedry. Vestibular system part 2: Psychophysics, applied aspects and general interpretations. In H. H. Kornhuber, editor, *Handbook of Sensory Physiology*, chapter Psychophysics of Vestibular Sensation, pages 3–154. Springer, 1974.
- [22] S. Han, M. Song, and J. Kwahk. A systematic method for analyzing magnitude estimation data. *International Journal of Industrial Ergonomics*, 23(5-6):513–524, 1999.
- [23] S. Hare, A. Saffari, and P. H. S. Torr. Struck: Structured output tracking with kernels. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 263–270, 2011.
- [24] R. Hartley and A. Zisserman. Two-view geometry. In *Multiple View Geometry in Computer Vision*, pages 237–361. Cambridge University Press, second edition edition, 2004.
- [25] M. Heilig. Sensorama simulator. US Patent Application, No. 3,050,870, Aug. 1962.
- [26] K. Hirota, S. Ebisawa, T. Amemiya, and Y. Ikei. A system for creating the content for a multi-sensory theater. In *Proceedings of the Virtual and Mixed Reality (Held as Part of HCI International)*, pages 151–157, 2011.

-
- [27] M. Hollins, R. Faldowski, S. Rao, and F. Young. Perceptual dimensions of tactile surface texture: A multidimensional scaling analysis. *Perception & Psychophysics*, 54(6):697–705, 1993.
- [28] K. Hong, J. Lee, and S. Choi. Demonstration-based vibrotactile pattern authoring. In *Proceedings of the International Conference on Tangible, Embedded and Embodied Interaction (TEI)*, pages 219–222, 2013.
- [29] W. Hutchinson and L. Knopoff. The acoustic component of western consonance. *Journal of New Music Research*, 7(1):1–29, 1978.
- [30] I. Hwang and S. Choi. Perceptual space and adjective rating of sinusoidal vibrations perceived via mobile device. In *Proc. Haptics Symposium*, pages 1–8. IEEE, 2010.
- [31] I. Hwang and S. Choi. Improved haptic music player with auditory saliency estimation. In *Lecture Notes on Computer Science (Eurohaptics)*, volume LNCS 8618, pages 232–240, 2014.
- [32] I. Hwang, H. Lee, and S. Choi. Real-time dual-band haptic music player for mobile devices. *IEEE Transactions on Haptics*, 6(3):340–351, 2013.
- [33] A. Israr and I. Poupyrev. Tactile brush: Drawing on skin with a tactile grid display. In *Proceedings of ACM CHI*, pages 2019–2028, 2011.
- [34] A. Israr, S. Zhao, K. Schwalje, R. Klatzky, and J. Lehman. Feel effects: Enriching storytelling with haptic feedback. *ACM Transactions on Applied Perception*, 2(3):1–17, 2014.
- [35] A. Kameoka and M. Kuriyagawa. Consonance theory, part I: Consonance of dyads. *The Journal of the Acoustical Society of America*, 45(6):1451–1459, 1969.

-
- [36] A. Kameoka and M. Kuriyagawa. Consonance theory, part II: Consonance of complex tones and its calculation method. *The Journal of the Acoustical Society of America*, 45(6):1460–1469, 1969.
- [37] M. Karam, F. A. Russo, and D. I. Fels. Designing the model human cochlea: An ambient crossmodal audio-tactile display. *IEEE Transactions on Haptics*, 2(3):160–169, 2009.
- [38] J. Kim, C.-G. Lee, Y. Kim, and J. Ryu. Construction of a haptic-enabled broadcasting system based on the MPEG-V standard. *Signal Processing: Image Communication*, 28(2):151–161, 2013.
- [39] M. Kim, S. Lee, and S. Choi. Saliency-driven real-time video-to-tactile translation. *IEEE Transactions on Haptics*, 7(3):394–404, 2014.
- [40] S. Kim, J. Kim, and K. Kim. Traveling vibrotactile wave - a new vibrotactile rendering method for mobile devices. *IEEE Transactions on Consumer Electronics*, 55(3):1032–1038, 2009.
- [41] S.-K. Kim. Authoring multisensorial content. *Signal Processing: Image Communication*, 28(2):162–167, 2013.
- [42] Y. Kim, J. Cha, J. Ryu, and I. Oakley. A tactile glove design and authoring system for immersive multimedia. *IEEE Multimedia*, 17(3):34–45, 2010.
- [43] R. L. Klatzky and S. J. Lederman. Tactile roughness perception with a rigid link interposed between skin and surface. *Perception & Psychophysics*, 61(4):591–607, 1999.

- [44] M. Konyo, S. Tadokoro, A. Yoshida, and N. Saiwaki. A tactile synthesis method using multiple frequency vibrations for representing virtual touch. In *Proc. IROS*, pages 3965–3971, 2005.
- [45] A. Krull, F. Michel, E. Brachmann, S. Gumhold, S. Ihrke, and C. Rother. 6-dof model based tracking via object coordinate regression. In *Proc. ECCV*, pages 384–399, 2015.
- [46] M. H. Kutner, C. J. Nachtsheim, J. Neter, and W. Li. *Applied Linear Statistical Models*. McGraw-Hill, 5th edition, 2005.
- [47] B. Lee, J. Lee, J. Cha, C. Seo, and J. Ryu. Immersive live sports experience with vibrotactile sensation. In *Human-Computer Interaction - INTERACT*, volume LNCS 3585, pages 1042–1045, 2005.
- [48] I. Lee and S. Han. Estimating the 3d posture of 6dof platform from image sequences. In *Proceedings of the International Symposium on Advanced Intelligent Systems*, pages 894–897, 2007.
- [49] J. Lee, J. Ryu, and S. Choi. Vibrotactile Score: A score metaphor for designing vibrotactile patterns. In *Proc. WHC*, pages 302–307. IEEE, 2009.
- [50] P. Lemmens, F. Cromptoets, D. Brokken, J. van den Eerenbeemd, and G.-J. de Vries. A body-conforming tactile jacket to enrich movie viewing. In *Proceedings of the Third Joint Eurohaptics Conference and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems (WorldHaptics)*, pages 7–12, 2009.
- [51] LG Electronics. Apparatus and method for generating vibration pattern. US Patent Application, No. 12/840,988, Jul. 21, 2010.

- [52] K. A. Li, T. Sohn, S. Huang, and W. G. Griswold. PeopleTones: A system for the detection and notification of buddy proximity on mobile phones. In *Proc. MobiSys*, pages 160–173. ACM Press, 2008.
- [53] C. Liu. *Beyond Pixels: Exploring New Representations and Applications for Motion Analysis*. PhD thesis, Massachusetts Institute of Technology, May 2009.
- [54] M. Lombard and T. Ditton. At the heart of it all: The concept of presence. *Journal of Computer-Mediated Communication*, 3(2), 1997.
- [55] M. Mainberger, A. Bruhn, and J. Weickert. Is dense optic flow useful to compute the fundamental matrix? In *Proceedings of International Conference on Image Analysis and Recognition*, volume LNCS 5112, pages 630–639, 2008.
- [56] M. T. Marshall and M. M. Wanderley. Examining the effects of embedded vibrotactile feedback on the feel of a digital musical instrument. In *Proc. NIME*, pages 399–404, 2011.
- [57] H. Matsukura, T. Nihei, and H. Ishida. Multi-sensorial field display: Presenting spatial distribution of airflow and odor. In *Proceedings of the IEEE Virtual Reality*, 2011.
- [58] H. Matsukura, T. Yoneda, and H. Ishida. Smelling screen: Development and evaluation of an olfactory display system for presenting a virtual odor source. *IEEE Transactions on Visualization and Computer Graphics*, 19(4):606–615, 2013.
- [59] T. Moon and G. J. Kim. Design and evaluation of a wind display for virtual reality. In *Proceedings of ACM VRST*, pages 122–128, 2004.
- [60] M. A. Nahon and L. D. Reid. Simulator motion-drive algorithms: A designer’s perspective. *Journal of Guidance, Control, and Dynamics*, 13(2):356–362, 1990.

- [61] T. Nakamoto and H. P. D. Minh. Improvement of olfactory display using solenoid valves. In *Proceedings of IEEE VR*, pages 179–186, 2007.
- [62] L. Nehaoua, H. Mohellebi, A. Amouri, H. Arioui, S. Espie, and A. Kheddar. Design and control of a small-clearance driving simulator. *IEEE Transactions on Vehicular Technology*, 57(2):736–746, 2008.
- [63] E. Oh, M. Lee, and S. Lee. How 4D effects cause different types of presence experience? In *Proceedings of the International Conference on Virtual Reality Continuum and Its Applications in Industry*, pages 375–378, 2011.
- [64] S. OModhrain and I. Oakley. Touch tv: Adding feeling to broadcast media. In *Proceedings of the European Conference on Interactive Television*, 2003.
- [65] R. V. Parrish, J. E. Dieudonne, and R. L. Bowle. Coordinated adaptive washout for motion simulators. *Journal of Aircraft*, 12(1):44–50, 1975.
- [66] K. Pauwels, L. Rubio, J. Diaz, and E. Ros. Real-time model-based rigid object pose estimation and tracking combining dense and sparse visual cues. In *Proc. CVPR*, pages 2347–2354, 2013.
- [67] N. A. Pouliot, C. M. Gosselin, and M. A. Nahon. Motion simulation capabilities of three-degree-of-freedom flight simulators. *Journal of Aircraft*, 35(1):9–17, 1998.
- [68] M. A. Rahman, A. Alkhalidi, J. Cha, and A. E. Saddik. Adding haptic feature to youtube. In *Proceedings of the International Conference on Multimedia*, pages 1643–1646, 2010.

-
- [69] L. D. Reid and M. A. Nahon. Flight simulation motion-base drive algorithms: Part1 - developing and testing equations. Technical report, Institute for Aerospace Studies, University of Toronto, 1985.
- [70] G. Reymond and A. Kemeny. Motion cueing in the renault driving simulator. *Vehicle System Dynamics*, 34(4):249–259, 2000.
- [71] D. P. Robertson and R. Cipolla. Structure from motion. In M. Varga, editor, *Practical Image Processing and Computer Vision*. John Wiley, 2009.
- [72] V. Rodehorst, M. Heinrichs, and O. Hellwich. Evaluation of relative pose estimation methods for multi-camera setups. In *Proceedings of the International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, pages 135–140, 2008.
- [73] E. Rosten and T. Drummond. Machine learning for high-speed corner detection. In *Proceedings of ECCV*, pages 430–443, 2006.
- [74] J. Ryu and S. Choi. posVibEditor: Graphical authoring tool of vibrotactile patterns. In *Proc. HAVE*, pages 120–125. IEEE, 2008.
- [75] J. Ryu, J. Jung, G. Park, and S. Choi. Psychophysical model for vibrotactile rendering in mobile devices. *Presence*, 19(4):364–387, 2010.
- [76] S. F. Schmidt and B. Conrad. Motion drive signals for piloted flight simulators. Technical Report CR-1601, NASA, 1970.
- [77] J. Seo and S. Choi. Perceptual analysis of vibrotactile flows on a mobile device. *IEEE Transactions on Haptics*, 6(4):522–527, 2013.

- [78] C. E. Sherrick and R. Rogers. Apparent haptic movement. *Perception and Psychophysics*, 1(3):175–180, 1966.
- [79] S. Shin, B. Yoo, and S. Han. A framework for automatic creation of motion effects from theatrical motion pictures. *Multimedia Systems*, 2013.
- [80] R. Sivan, J. Ish-Shalom, and J.-K. Huang. An optimal control approach to the design of moving flight simulators. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-12(6):818–827, 1982.
- [81] P. N. Vassilakis. SRA: A web-based research tool for spectral and roughness analysis of sound signals. In *Proc. SMC*, pages 319–325, 2007.
- [82] R. T. Verrillo. Measurement of vibrotactile sensation magnitude. In S. J. Bolanowski and G. A. Gescheider, editors, *Ratio Scaling of Psychological Magnitude: In Honor of the Memory of S. S. Stevens*, pages 260–275. Lawrence Erlbaum Associates, Inc., 1991.
- [83] M. von der Heyde and B. E. Riecke. How to cheat in motion simulation - comparing the engineering and fun ride approach to motion cueing. Technical Report 089, Max-Planck-Institut, 2001.
- [84] M. Walzl, B. Rainer, C. Timmerer, and H. Hellwagner. An end-to-end tool chain for sensory experience based on MPEG-V. *Signal Processing: Image Communication*, 28(2):136–150, 2013.
- [85] M. Werlberger, W. Trobin, T. Pock, A. Wedel, D. Cremers, and H. Bischof. Anisotropic Huber-L1 optical flow. In *Proceedings of BMVC*, pages 1–7, 2009.

-
- [86] G. Whitehead. The technological art of simulation. *SMPTE Motion Imaging*, 110:39–42, 2001.
- [87] Y. Wu, J. Lim, and M.-H. Yang. Online object tracking: A benchmark. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2411–2418, 2013.
- [88] A. Yamashita. Theaters of illusion: The continuing evolution of entertainment simulation. *ACM SIGGRAPH Computer Graphics*, 28(2):142–144, 1994.
- [89] H. Yao, D. Grant, and J. M. Cruz-Hernandez. Perceived vibration strength in mobile devices: The effect of weight and frequency. *IEEE Transactions on Haptics*, 3(1):56–62, 2010.
- [90] Y. Yoo, I. Hwang, and S. Choi. Consonance perception of vibrotactile chords: A feasibility study. In *Proc. HAID*, volume LNCS 6851, pages 42–51, 2011.
- [91] K. Yoon, B. Choi, E.-S. Lee, and T.-B. Lim. 4-D broadcasting with MPEG-V. In *Proceedings of the IEEE International Workshop on Multimedia Signal Processing*, pages 257–262, 2010.
- [92] J. Zhang, S. Ma, and S. Sclaroff. Meem: Robust tracking via multiple experts using entropy minimization. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 188–203, 2014.
- [93] S. Zhao, O. Schneider, R. Klatzky, J. Lehman, and A. Israr. Feelcraft: Crafting tactile experiences for media using a feel effect library. In *Proceedings of the ACM User Interface Software and Technology Symposium (UIST)*, 2014.

-
- [94] W. Zhong, H. Lu, and M.-H. Yang. Robust object tracking via sparsity-based collaborative model. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1838–1845, 2012.
- [95] E. Zwicker and H. Fastl. *Psychoacoustics: Facts and models*. Springer-Verlag, 2nd edition, 1999.

Acknowledgements

감사의 글

공부와 연구, 논문 작성 뿐 아니라 제 인생에 있어서도 좋은 선생님과 선배가 되어 주신 지도교수님 최승문 교수님께 먼저 큰 감사를 드립니다. 그리고 바쁘신 가운데도 제 비전 부분 연구에 적극적인 조언을 해주시고 진로에도 많은 도움을 주셨던 한보형 교수님께 감사 드립니다. 제 박사 연구와 논문을 심사해주시고 좋은 지적을 해주셨던 이승용, 홍기상, 이성길 교수님께도 감사 드립니다. 많은 교수님들의 지도가 없었더라면 제 박사 연구는 있을 수 없었을 것입니다.

석사 때부터 늘 좋은 조언자가 되어주었던 선배 석희형으로부터 성훈이 형, 인욱이, 인이, 건혁이, Reza, 제게 여러가지로 많은 도움을 주고 말벗이 되어주어 고맙습니다. 후배들인 종만이, 호진이, 성환이, 용재, 명찬이, 경표, 호준이, 준석이, 한슬이에게는 무엇보다 제가 인격적으로 본이 될 만한 모습을 많이 보이지 못하고 그들의 공부와 연구에 많은 도움을 주지 못한 것 때문에 정말 미안한 마음 뿐입니다. 꼭 연구가 빨리 그리고 알차게 잘 마무리 되어 각자 원하는 좋은 길을 잘 찾아갈 수 있으면 좋겠습니다.

2001년에 학부생으로 포항에 처음와서 중간에 떠나 있었던 시간을 제외해도 10년의 시간을 포항에서 보냈습니다. 그 동안 제가 누구보다 좋아하고 사랑했던 친구들, 선배들, 후배들, 교수님들이 너무 많아 여기에 다 적을 수가 없습니다. 돌아보면 힘든 과정도 있었지만 다 행복하고 좋은 추억이었고 제 미래를 만들어주는데 정말 소중한 시간이었습니다. 그 때 함께 해주었던 많은 사람들 덕분입니다. 무엇보다 효자 교회와 성도들, 학생들과 함께 한 시간은 제 포항 생활에서 가장 행복한 시간들이었습니다.

마지막으로 언제나 저를 믿고 지원을 아끼지 않으셨던 아버지, 어머니, 장인어른,

장모님께 감사 드립니다. 그동안 많이 고생하면서 불확실한 미래에도 불구하고 늘 저를 잘 따라주었던 인생의 동반자, 동역자 아내 누리 고마워요 여보. 내 행복과 기쁨인 다은이 다현이. 이제까지 지켜주시고 힘과 도움이 되어주신 하나님 감사합니다.

Curriculum Vitae

Name : Jaebong Lee

Date of Birth : 1982. 5. 13

Present Address : 경북 포항시 남구 청암로 77 대학원아파트 2동 101호

Education

2001–2008 : Department of Mathematics, POSTECH (B.S.)

2008–2010 : Department of Computer Science and Engineernig, POSTECH (M.S.)

2012–2016 : Department of Computer Science and Engineernig, POSTECH (Ph.D.)

Thesis Title :

4D 영화를 위한 모션 및 진동 효과의 자동 생성

(Motion and Vibration Effects Synthesis for 4D Films)

Advisor: Prof. Seungmoon Choi

Experience

2010–2012 : Software Research Engineer, LG Electronics Inc., Seoul, Korea

Affiliation

HVR Lab., Department of Computer Science and Engineering, POSTECH

Publications

International Journals

1. **Jaebong Lee**, Bohyung Han, and Seungmoon Choi, "Motion Effects Synthesis for 4D Films," *IEEE Transactions on Visualization and Computer Graphics* (accepted).

International Conferences

1. Hoang Minh Phuong, **Jaebong Lee**, Hojin Lee, Kyusong Lee, Gary Geunbae Lee, and Seungmoon Choi, "Haptic-enabled English Education System," In *Proceedings of the Asia Haptics*, 2014.
2. **Jaebong Lee**, Eunji Cho, Minjae Kim, Yongmin Yoon, and Seungmoon Choi, "PreventFHP: Detection and Warning System for Forward Head Posture," In *Proceedings of the IEEE Haptics Symposium (HS)*, pp. 295-298, 2014 (Winner of the Best Teaser Award).
3. **Jaebong Lee** and Seungmoon Choi, "Real-time Perception-Level Translation from Audio Signals to Vibrotactile Effects," In *Proceedings of the ACM SIGCHI Conference on Human factors in Computing Systems (CHI)*, pp. 2567-2576, 2013 (Acceptance rate = 20%).
4. Kyungpyo Hong, **Jaebong Lee**, and Seungmoon Choi, "Demonstration-Based Vibrotactile Pattern Authoring," In *Proceedings of the ACM International Conference on Tangible, Embedded and Embodied Interaction (TEI)*, pp. 219-222, 2013 (Acceptance rate = 35%).

5. **Jaebong Lee** and Seungmoon Choi, "Evaluation of Vibrotactile Pattern Design Using Vibrotactile Score," In *Proceedings of the IEEE Haptics Symposium (HS)*, pp. 231-238, 2012 (Long oral presentation; Acceptance rate = 26%).
6. **Jaebong Lee** and Seungmoon Choi, "Effects of Haptic Guidance and Disturbance on Motor Learning: Potential Advantage of Haptic Disturbance," In *Proceedings of the IEEE Haptics Symposium (HS)*, pp. 335-342, 2010 (Nominee for Best Paper Award; Oral presentation; Acceptance rate = 29.5%).
7. Gabjong Han, **Jaebong Lee**, In Lee, Seokhee Jeon, and Seungmoon Choi, "Effects of Kinesthetic Information on Working Memory for 2D Sequential Selection Task," In *Proceedings of the IEEE Haptics Symposium (HS)*, pp. 43-46, 2010 (Oral presentation; Extended abstract; Acceptance rate = 18.7%).
8. **Jaebong Lee**, Jonghyun Ryu, and Seungmoon Choi, "Vibrotactile Score: A Score Metaphor for Designing Vibrotactile Patterns," In *Proceedings of the IEEE World Haptics Conference (WHC)*, pp. 302-307, 2009.
9. **Jaebong Lee** and Seungmoon Choi, "Haptic Pottery Modeling System Using Improved Circular Sector Element Method," In *Proceedings of the International Conference on Mechatronics and Information Technology (ICMIT)*, pp. 7-9, 2009.
10. **Jaebong Lee**, Gabjong Han, and Seungmoon Choi, "Haptic Pottery Modeling Using Circular Sector Element Method," *Lecture Notes on Computer Science (EuroHaptics)*, vol. 5024, pp. 668-674, 2008.

Papers In Preparation

1. **Jaebong Lee**, Bohyung Han, and Seungmoon Choi, "Interactive Motion Effects Design for a Moving Object in 4D films".

Domestic Journals

1. **Jaebong Lee**, Kyusong Lee, Hoang Minh Phuong, Hojin Lee, Gary Geunbae Lee, and Seungmoon Choi, "POMY: POSTECH Immersive English Study with Haptic Feedback," *Journal of Institute of Control, Robotics and Systems*, vol. 20, no. 8, pp. 815-821, 2014.
2. **Jaebong Lee**, Gabjong Han, and Seungmoon Choi, "A Haptic Pottery Modeling System Using GPU-Based Circular Sector Element Method," *Journal of KIISE: Software and Application*, vol. 37, no. 8, pp. 611-619, 2010.

Domestic Conferences

1. **Jaebong Lee** and Seungmoon Choi, "Automatic Generation of 4D Motion Effects," In *Proceedings of the HCI Korea*, pp. 29-31, 2014 (Winner of the best paper award).
2. **Jaebong Lee** and Seungmoon Choi, "Immersive Computer-Assisted Language Learning Using Haptic Chair," In *Proceedings of the KROC*, pp. 360-361, 2013.
3. **Jaebong Lee** and Seungmoon Choi, "Improvement of the Haptic Pottery Modeling System Using Circular Sector Element Method," In *Proceedings of Next Generation Computing Fall Conference*, 2009.

4. **Jaebong Lee**, In Lee, Gabjong Han, Seokhee Jeon and Seungmoon Choi, "Effect of Haptic Sensory Information on Short-term Memory Chunking in 2D Sequential Selection Task," In *Proceedings of the KROC*, pp. 455-457, 2009.
5. **Jaebong Lee**, Gabjong Han and Seungmoon Choi, "Pottery Modeling Using Circular Sector Element Method," In *Proceedings of the HCI Korea*, Vol. 1, pp. 78-84, 2008.

Demonstrations

1. **Jaebong Lee**, Eunji Cho, Minjae Kim, Yongmin Yoon, and Seungmoon Choi, "PreventFHP: Detection and Warning System for Forward Head Posture," Demonstrated in *the IEEE Haptics Symposium (HS)*, 2014.
2. Kyusong Lee, **Jaebong Lee**, Nohkyung Lee, Chiyoung Lee, Hoang Minh Phuong, Sangdo Han and Hojin Lee, "POMY (POSTECH Immersive English Study)," Demonstrated in *the HCI Korea*, 2014 (Winner of the best HCI KIDS award).
3. **Jaebong Lee** and Seungmoon Choi, "Vibrotactile Applications for Mobile Devices," Demonstrated in *the Microsoft Asia Faculty Summit*, 2012.
4. **Jaebong Lee**, Jonghyun Ryu, and Seungmoon Choi, "Graphical Authoring Tools for Vibrotactile Patterns," In *DVD Proceedings of World Haptics Conference (WHC)*, pp. 388-389, 2009.

Patents

1. Seungmoon Choi and **Jaebong Lee**, "Apparatus and Method for Generating Motion Effects by Analyzing Motion of Object," Korea Patent Pending, No. 10-2014-0184271, December 19, 2014.
2. Seungmoon Choi and **Jaebong Lee**, "Apparatus for Generating Motion Effects and Computer Readable Medium for the Same," Korea Patent Pending, No. 10-2014-0178616, December 11, 2014.
3. Eunhwa Lee, Seungmoon Choi, **Jaebong Lee**, Yongjae Yoo, Jeongseok Lee, Daekwang Jung, and Yudong Bae, "Method and Device for Generating Vibration From Adjective Space," Korea Patent Pending, No. 10-2014-0012213, February 3, 2014.
4. Eunhwa Lee, Seungmoon Choi, **Jaebong Lee**, Jeongseok Lee, Daekwang Jung, Yudong Bae, Jongman Seo, Yongjae Yoo, and Jaemin Chun, "Method and Device for Generating Vibration Using Adjective," Korea Patent Pending, No. 10-2014-0010881, January 28, 2014.
5. Seungmoon Choi and **Jaebong Lee**, "Apparatus and Method for Providing Motion Haptic Effect Using Video Analysis," U.S. Patent Pending, No. 14/518238, Korea Patent Pending, No. 10-2013-0125156, October 21, 2013.
6. Seungmoon Choi, Minjae Kim, DaiJin Kim, Wankyung Chung, **Jaebong Lee**, Eunji cho, and Yongmin Yoon, "System for Warning Forward Head Posture and Method Thereof," Korea Patent No. 10-2014-0110173, March 5, 2013.
7. Seungmoon Choi and **Jaebong Lee**, "Method of Converting Audio signal to

Haptic Signal and Apparatus Thereof,” U.S. Patent Pending, No. US 2014/0167940 A1, Korea Patent No. 10-2014-0079582, December 17, 2012.

8. Seungmoon Choi, **Jaebong Lee**, and Kyungpyo Hong, “Method for Generating Vibration Pattern and for the Same,” Korea Patent No. 10-2013-0113191, April 5, 2012.
9. Hyunsun Hong, Sora Kang, **Jaebong Lee**, Munchae Joung, “Haptic Transmission Method and Mobile Terminal for Same,” U.S. Patent Pending, No. WO2013089294 A1, Korea Patent Pending, No. 10-2014-0092837, December 15, 2011.
10. Kyunghun Hwang, Hyunsun Hong, **Jaebong Lee**, Munchae Joung, Sunuk Kim, “An Apparatus for Controlling Residual Vibration,” Korea Patent Pending, No. 10-2012-0130471, May 23, 2011.
11. **Jaebong Lee**, Junghwan Kim, Sunuk Kim, Kyunghun Hwang, Munchae Joung, and Hyunsun Hong, “An Apparatus for Generating the Vibrating Feedback From Input Audio Signal,” Korea Patent Pending, No. 10-2012-0126446, May 11, 2011.
12. Seungmoon Choi, Jonghyun Ryu, and **Jaebong Lee**, “Vibration Authoring Tool, Vibration Authoring Method, and Storage Medium Recorded With the Same,” U.S. Patent No. 7,999,166 B2, Korea Patent No. 10-2010-0125219, March 13, 2009.