Attention-guided Algorithms to Retarget and Augment Animations, Stills, and Videos

Eakta Jain

TR-12-14

May 13, 2012

The Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy.

Abstract

Still pictures, animations and videos are used by artists to tell stories visually. Computer graphics algorithms create visual stories too, either automatically, or, by assisting artists. Why is it so hard to create algorithms that perform like a trained visual artist? The reason is that artists think about where a viewer will look at and how their attention will flow across the scene, but algorithms do not have a similarly sophisticated understanding of the viewer.

Our key insight is that computer graphics algorithms should be designed to take into account how viewer attention is allocated. We first show that designing optimization terms based on viewers' attentional priorities allows the algorithm to handle artistic license in the input data, such as geometric inconsistencies in hand-drawn shapes. We then show that measurements of viewer attention enables algorithms to infer high-level information about a scene, for example, the object of storytelling interest in every frame of a video.

All the presented algorithms retarget or augment the traditional form of a visual art. Traditional art includes artwork such as printed comics, i.e., pictures that were created before computers became mainstream. It also refers to artwork that can be created in the way it was done before computers, for example, hand-drawn animation and live action films. Connecting traditional art with computational algorithms allows us to leverage the unique strengths on either side. We demonstrate these ideas on three applications:

Retargeting and augmenting animations: Two widely practiced forms of animation are two-dimensional (2D) hand-drawn animation and three-dimensional (3D) computer animation. To apply the techniques of the 3D medium to 2D animation, researchers have attempted to compute 3D reconstructions of the shape and motion of the hand-drawn character, which are meant to act as their 'proxy' in the 3D environment. We argue that a perfect reconstruction is excessive because it does not leverage the characteristics of viewer attention. We present algorithms to generate a 3D proxy with different levels of detail, such that at each level the error terms account for quantities that will attract viewer attention. These algorithms allow a hand-drawn animation to be retargeted to a 3D skeleton and be augmented with physically simulated secondary effects.

Augmenting stills: Moves-on-stills is a technique to engage the viewer while presenting still pictures on television or in movies. This effect is widely used to augment comics to create 'motion comics'. Though state of the art software, like iMovie, allows a user to specify the parameters of the camera move, it does not solve the problem of how the parameters are chosen. We believe that a good camera move respects the visual route designed by the artist who crafted the still picture; if we record the gaze of viewers looking at composed still pictures, we can reconstruct the artist's intention. We show, through a perceptual study, that the artist succeeds in directing viewer attention in comic book pictures, and we present an algorithm to predict the parameters of camera moves-on-stills from statistics derived from eyetracking data.

Retargeting video: Video retargeting is the process of altering the original video to fit the new display size, while best preserving content and minimizing artifacts. Recent techniques define content as color, edges, faces and other image-based saliency features. We suggest that content is, in fact, what people look at. We introduce a novel operator that extends the classic "pan-and-scan" to introduce cuts in addition to automatic pans based on viewer eyetracking data. We also present a gaze-based evaluation criterion to quantify the performance of our operator.

Acknowledgments

This thesis would not have been possible without my advisers, Jessica Hodgins and Yaser Sheikh. Thank you for your guidance and support.

Thank you also to my committee, Nancy Pollard and Adam Finkelstein, for shaping the course of this work. Moshe Mahler, I enjoyed working with you on all the projects. Arik Shamir, thank you for the insights and thoughtful discussions.

To my friends, office-mates and colleagues in the CMU Graphics Lab, the robotics graduate program and Disney Research Pittsburgh: thanks for all the long lunches, painstaking annotations and wonderful memories.

To all the artists who let me use their work as input to my algorithms: it was a pleasure to compute with such beautiful things.

To my family: thanks for cheering me on.

To my husband: thanks for your patience. And also all the chocolate ice cream.

Contents

Chapter 1

Introduction

Often in a comic book, we see pictures of superheros with impossibly large muscles strike a physically undoable pose as they slam their enemy. Or we watch events on either side of an opaque wall in quick succession as part of a movie, even though a physical viewer could not possibly have traveled from one room to the other that fast. Or we see a cartoon character sit on a pin and then jump much higher than gravity permits. These are examples of content created by an artist to tell a story. Contrast them with 'captured' visual content, such as, the feed from a security camera. Though both forms of visual content are pixels on a screen, for artist-created content expressing emotion and conveying the narrative are the important goal, not maintaining the constraints of realism, such as geometry or gravity [77]. Thus, it is challenging to model the components of artist-created content: because they involve not only physics, but also cognition.

Researchers are making some headway with models of cognition both for the artist and the viewer. Recent studies have examined the process of hand drawing [17, 84] and there is a large body of ongoing work on how viewers perceive computer-generated graphics [80]. This work includes psychophysical experiments on the perception of human motion [71, 73, 74, 75], physically based phenomena such as clothing [72], laws of physics such as gravity [92, 94] and rigid body collisions [93], and the perception of complexity in a scene [88, 89, 90]. This research aims to come up with rules based on systematic observations of human subjects that can guide both human artists and computer algorithms in improving efficiency and quality.

While these models of cognition are being developed, there is also an increasing interest in collecting measurements from the artist and the viewer to drive algorithms. Researchers have motion-captured the artist/actor to more easily create character animations that match the artist's concept of the character [60, 130]. It is also possible to instrument the viewer to collect various types of data such as brain activation through fMRI or EEG, heartrate through an EKG sensor, or head movements and gaze through a person-facing camera. In particular, gaze data, which is the only measurable part of visual attention, has been used to drive algorithms for painterly rendering [24] and image cropping [106].

Visual attention is a key variable in manipulating artist-created content because many sophisticated decisions made by the artist and the viewer are observable in how the viewer's attention moves through the scene. We are motivated by the observation that visual artists actively manage viewer attention as part of creating their content. From the time of the Renaissance painters, 'rules of good painting' like spatial layout and figure composition direct the viewer to look towards a particular person, object or action in the scene. More recently, in 1985, Will Eisner suggested that "...In sequential art the artist must, from the outset, secure control of the reader's attention and dictate the sequence in which the reader will follow the narrative...the most important obstacle to surmount is the tendency of the reader's eye to wander..." [32]. The medium of film allows the artist to craft both the spatial composition inside the frame and the temporal order of presenting information to direct the viewers' focus of interest, or, "eye-trace" [77]. **In this thesis, we show that viewer attention is a handle that algorithms may use to retarget and augment artist-created animations, still pictures and videos in two ways: through guidelines to design an optimization and as recorded eyetracking data that is input to an algorithm.**

The first form of visual content that we study is animation. Traditional two-dimensional (2D) animation and computer-generated three-dimensional (3D) animation have developed almost independently of one another because the two media have many fundamental differences. The animated character is a 2D shape in the former, while it is a 3D geometric object in the latter. The process of traditional 2D animation consists of drawing shapes on paper (or a tablet) by hand. In contrast, 3D animators are trained to operate computer software, move skeletal rigs, and specify joint angles. There is some evidence that viewers experience the two types of animation differently. For example, John Lasseter points out that 'held drawings' are common in 2D animation but not in 3D computer animation because a hold in 3D causes "...the motion to die..." [62]. As a result, techniques developed for one medium of animation do not immediately carry over to the other medium. Connecting 3D animation and 2D animation would allow the 2D medium to leverage the strengths of 3D CG animation, such as physical simulation.

The natural way to create a connection between the two media would be to create a 3D geometric object that is a 'proxy' for the hand-drawn character in the 3D environment. However, creating this 3D proxy with consistent geometry may be impossible because hand animators routinely introduce inconsistencies in the shapes they draw to add expressiveness to the character [86]. We propose the notion of different levels of detail in a 3D proxy to connect 3D animation with hand animation: at each level, the proxy preserves what the viewer might notice. For example, if we want to attach simulated pompoms (a 3D physical simulation) to the wrists of a hand-drawn character, the 3D proxy should accurately track the hand-drawn character's wrists in the image plane; otherwise, the simulated pompoms would appear to leave the wrists of the hand-drawn character. Creating a 3D proxy that attempts to reconstruct full 3D pose, minimizes an objective function with rigidity constraints, or enforces ground contact for the feet, would not only be excessive, but might even introduce error in the wrist at the expense of minimizing extraneous error terms. The first contribution of this thesis is to show that the different levels of detail can be generated by reinterpreting three basic error terms based on what errors will draw viewer attention at each level of detail. In Chapter 2, we describe the input-matching term, the motion prior term and the smoothing term, and present results on a variety of animated sequences.

The second form of visual content that we investigate is still pictures created by artists, specifically, comic art. Moves-on-stills is a technique to engage the viewer while presenting still pictures on television or in movies. This effect is used to create 'motion comics' from comic book material. State of the art software, like iMovie, allow a user to specify the parameters of the camera move; however, they do not solve the problem of how the parameters are chosen. We argue that a good move should respect the flow of visual attention intended by the artist who designed the still picture. Such a move would support the narrative in the still pictures. The challenge in extracting this visual route is that we cannot compute artistic intent from image features alone: computational algorithms to predict saliency cannot yet recognize all contextual cues, infer relationships between objects or understand narrative.

We propose to use the information in recorded viewer gaze data to reconstruct the artist-intended visual route in a comic book panel. Even though viewer attention is influenced by both stimuli characteristics and person-specific idiosyncrasies, the visual route created by the artist dominates over individual idiosyncrasies. The second contribution of this thesis is to verify experimentally that there is increased agreement in the gaze data of people looking at comic art versus a baseline of random photographs, and to demonstrate an algorithm to predict the parameters of camera moves-on-stills based on aggregate statistics derived from the recorded gaze data of multiple viewers. In Chapter 3, we explain the details of our experiments with human participants and show the results of our algorithm on several comic book panels. We compare the moves-on-stills created by our algorithm with a professionally created motion comic.

The third form of visual content that we study is moving pictures or film/video data. Digital video is easy to store and transport, and as a result, can be marketed to a global audience on a variety of platforms, ranging from theater screens to smartphones. However, it needs to be altered to be accessible to this wide audience. Most current methods retarget original video to a different aspect ratio by rescaling or cropping the original video. All methods agree that it is important to preserve the 'important' regions in the original video and rely on computed measures such as contrast or edge density to identify the importance of a region. These measures often misfire because they do not understand high-level information such as narrative, and do not process audio.

We propose that instead of performing intermediate tasks, such as object recognition, scene context and narrative understanding, with the goal of identifying 'important' regions in a video, we measure what is important in a video by recording what viewers attend to. Because filmmakers deliberately direct viewer gaze as part of creating the film through devices such as a loud noise or having the actor move [28], recorded viewer gaze is the gold standard for the regions that are important to preserve while manipulating the original video. As a result, we are able to expand the definition of video retargeting from scaling or cropping to *re-editing* the original video. The third contribution of this thesis is an algorithm to re-edit a widescreen video through the pan-and-scan operator (virtual camera moves to keep the object of interest onscreen) and the introduction of new cuts driven by eyetracking data from multiple viewers. In Chapter 4, we demonstrate our algorithm on a variety of clips taken from three Hollywood films. Further, we evaluate our results by comparing the difference in viewer gaze on the original and re-edited video. Thus, eyetracking data serves as a behavioral metric for evaluating the quality of re-edits performed by our method.

It is the technologist's challenge to connect new media with traditional media and preserve the visual experience intended by the artist in doing so. Viewer attention is a key behavioral cue to enable this connection. In the last twenty years, mainstream visual media has become largely digital in nature. This shift has occurred because digital content is more easily modifiable (when working with physical media, 'modify content' was often synonymous with 'redo content'), it is accessible to a wider audience because it is easier to store and transport, and many visually entertaining effects are more easily created digitally (for example, realistic tidal waves via computer graphics simulations, or a 'frosted glass effect' via procedural image filters). Sketching and painting on a stylus-tablet setup is now so ubiquitous that it is not surprising that the comic book *Spiderwoman: Agent of S.W.O.R.D.* was created entirely on the computer. *Toy Story 3* and *Avatar* are both computer generated movies. In 2002, *Star Wars Episode II: Attack of the Clones* was the first major motion picture to be shot entirely with a digital camera without film backup [95]. Since then, both high-budget live action cinema, and lower-budget television content has been increasingly shot with digital cameras.

Understanding what the viewer cares about helps focus algorithmic resources in the right places. Recording viewer gaze provides information about the high-level semantics of a scene. Comparing points of regard before and after a digital manipulation helps better preserve the original artist-crafted visual experience. With the launch of new eyetracking devices that can be head-mounted [10], worn as eyeglasses [27], embedded in a laptop [78] or in a smartphone

[11], it will become possible to record viewer attention in a natural setting. This information could then be incorporated into digital visual media. For example, a digital image could now be composed of four channels: the color channels RGB, and a fourth channel S that encodes visual saliency. A codec that understands this fourth channel could be downloaded and any display device would now become an attention-guided device, thus providing viewers with an experience that was as-similar-as-possible to the experience conceived by the artist who created the original visual content.

Chapter 2

Retargeting and Augmenting Hand Animations with Three-dimensional Proxies

2.1 Introduction

Depending on their training and personal preference, animators adopt many different media to make their characters move and emote—some favor sketching with a pencil, others like to manipulate armatures, sculpt clay, move paper cutouts, or pose 3D computer-generated skeletal rigs. Each medium has advantages and disadvantages, and requires that the artist develop a medium-specific set of skills to create a visually compelling end product [62].

The advantage of sketching with a pencil is that communicating the basic idea is quick; but art directing hand-drawn animations is cumbersome. If the director wants a change in how the lead character's skirt swishes, the animator has to redraw the skirt for every frame. In contrast, the medium of 3D computer animation makes many components easily modifiable—the clothes can be re-simulated, the viewing camera can be moved, and the lighting can be changed with little work on the part of the artist. This modifiability comes at the cost of increased initial effort to set up the scene and dynamic simulations, yet it enables 3D computer animation to be a more easily modifiable medium than traditional hand-drawn animation.

The skills involved in hand-drawn animation and 3D computer animation are different too. Traditional animators are trained to communicate the movement of the character through pencil strokes [54]. In contrast, 3D artists are skilled at using computer software and at manipulating curves, control points and inverse kinematics handles to construct a performance for their character.

These observations raise the question: can we connect these two different media so as to leverage the programmability of 3D computer animation in the hand-drawn medium, and the unique talent pool of traditional animators in the 3D computer-generated medium? The challenge here is that the elements of 3D computer animation and 2D hand-drawn animation are fundamentally incompatible; the character exists as a consistent 3D object in the former, while 2D artists routinely introduce inconsistencies in their drawings to add expressiveness to the character [86]. There may, in fact, be no consistent geometry that can be used to accurately describe a hand-drawn character from all points of view, or to define the imaginary camera used by the artist to create their drawing.

We propose that hand-drawn characters be represented by 3D proxies with different levels of detail: at each level, the error terms are reinterpreted to better preserve what is important to the viewer. In Figure 2.1, we arrange the 3D proxies that can be created for a given hand-drawn animation according to increasing level of detail. For example, if we wanted to attach a simulated balloon onto the wrists of a hand-drawn character, the level of detail for the 3D proxy would be a plausible 3D trajectory for a chosen point (wrist marker) on the 2D hand-drawn character—the single-point 3D proxy. Attempting to reconstruct the full 3D pose, minimizing an objective function with rigidity constraints, or enforcing good ground contact models would not only be excessive, but might also introduce error in the desired trajectory at the expense of minimizing extraneous error terms.

The next higher level of detail would be an approximate 3D model composed of tapered cylinders and spheres. These 3D proxies are well-suited to drive simulated effects like cloth and fluids, without having to fit exactly to the hand-drawn pencil strokes, because they will only provide the driving signal and will never be directly rendered. Here, the objective is to track points of contact precisely, and approximate the rest of the character with polygonal shapes—for

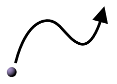
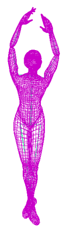
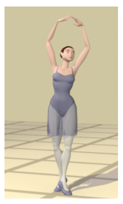
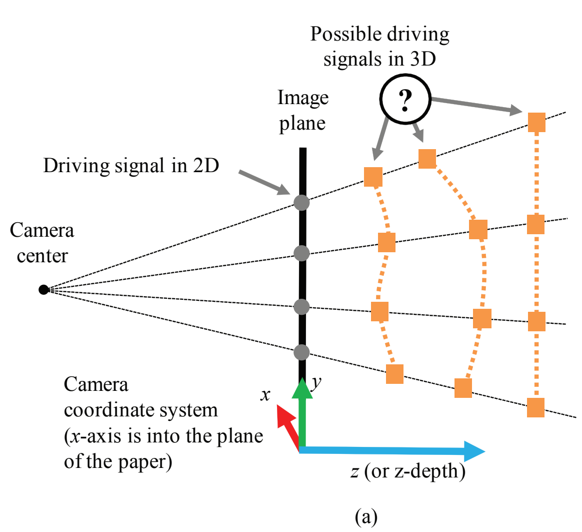
3D proxy	Degrees of freedom	Function
Single-point 	$3N$ N= number of attachment points (typically, $N = 1,2,3$)	Attachment point
3D Polygonal shapes	$5V$ V= number of tapered cylinders (typically, $V = 1$ to 25)	Collision volume
Joint hierarchy 	$3J$ J= number of joints (typically, $J = 20$)	Skinned, lit, rendered from any viewpoint 

Figure 2.1: Three-dimensional proxies for hand-drawn characters are arranged in increasing order of degrees of freedom. The first level is the single-point proxy—for example, the 3D trajectory of a single wrist marker required to hold a balloon. The next level is approximate cylindrical models for those body parts which interact with 3D scene elements. The third level is a hierarchical joint model, which can be rigged, skinned, lit, and placed in a 3D scene.

example, precisely tracking points on the wrist so that the scarves appear to be attached to the ballerina, but approximating the arms as tapered cylinders (Figures 2.1 and 2.14).

Finally, transferring the style of the hand-drawn animation onto a hierarchical 3D joint model would allow traditionally trained animators to control a classic skeleton-based 3D character—animators who think in terms of pencil lines would be able to puppeteer 3D characters that can also be rigged, skinned, lit and accessorized with simulated clothes. Here, smoothness and naturalness of motion take on more importance than accurately tracking image pixels.

In this chapter, we present an integrated approach to generate three levels of 3D proxies for a hand-animated character—the single-point proxy, the tapered cylinder model, and the hierarchical joint model. We frame the solution as a least squares minimization and show that by reformulating the terms of the minimization according to the level of detail desired in the 3D proxy, we can generate proxies for a hand-drawn character that better preserve qualities that the


Possible driving signals in 3D
?
Image plane
Driving signal in 2D
Camera center
Camera coordinate system (x-axis is into the plane of the paper)
x
y
z (or z-depth)
(a)

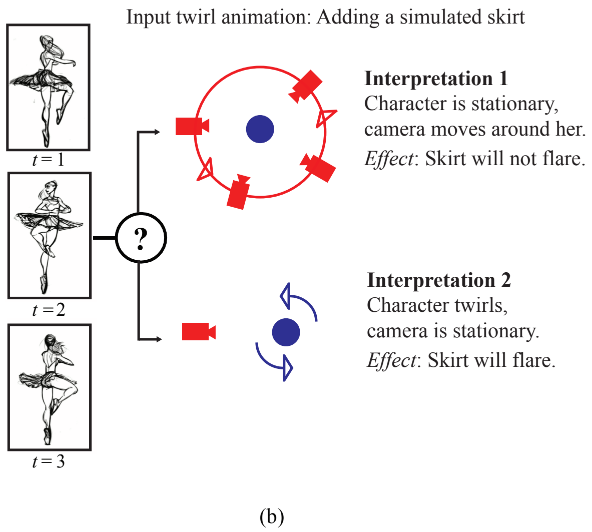

Input twirl animation: Adding a simulated skirt
t = 1
t = 2
t = 3
?
Interpretation 1
Character is stationary, camera moves around her.
Effect: Skirt will not flare.
Interpretation 2
Character twirls, camera is stationary.
Effect: Skirt will flare.
(b)

Figure 2.2: (a) Depth ambiguity: Multiple 3D trajectories can yield the same 2D projected path. (b) Composite motion ambiguity: The motion of the camera can not be disambiguated from the motion of the character if we are only given the image plane information.

viewer will care about in the final animation. Earlier versions of this work were published at the ACM/Eurographics Symposium on Computer Animation [50, 51]. Also, a longer journal paper, where we have generalized the technique and more extensively evaluate the various underlying assumptions, has been submitted to ACM Transactions on Graphics.

We have made certain design choices with the standard animation pipeline in mind—when choosing inputs for the system, we employ user interaction that can be reasonably integrated into the traditional animation workflow. We present results on a variety of hand-drawn animated characters, ranging from stick figures to a ballerina with human shaped limbs, and evaluate the system through synthetic tests and hand animations that stretch the underlying assumptions.

2.2 Background

We discuss past work on the use of 3D representations for 2D objects in the computer graphics (CG) context and the techniques used to generate them. There has also been a great deal of research in reconstructing 3D human pose from images and we briefly survey that body of work in the computer vision community.

2.2.1 Computer Graphics

One of the earliest forms of 3D proxy for a hand-drawn scene element was used in the movie *Spirit*—3D models of hand-animated horses were carefully made to match the artist's lines, and shots switched seamlessly between the hand-drawn elements and the 3D elements depending on whether the characters were seen in closeup or long shots [19].

Petrovic and colleagues [83] create a blobby 3D proxy to generate ray-traced shadows for a 2D character by inflating the 2D shape along the third dimension. Their main contribution is a graphical interface that allows the user to specify the relative depths of every scene element that needs to be inflated. Correa and colleagues [20] warp a 3D mesh model to match the artist-drawn shape—the mesh model can then be re-lit and textured and rendered to fill in the insides of the artist's hand-drawn shape with an intricate pattern. Johnston [55] bypasses a full 3D proxy in favor of only interpolating normals, a 2.5D proxy that is sufficient to integrate 3D lighting with a hand drawing. All these methods are designed to work with blobby shapes and do not maintain articulation constraints.

There are also several previous techniques for lifting hand-drawn articulated characters from 2D to 3D. All of these techniques deal with various ways to resolve the forward-backward depth ambiguity. For example, Davis and colleagues [23] sort the multiple 3D interpretations of a 2D pose according to joint angle constraints and other heuristics, but leave the final selection to the user. Their goal is to simplify the process of creating a 3D pose—the artist needs to know how to draw and mouse-click, but need not be trained in rigging and manipulating inverse kinematics handles. The generated 3D pose can be thought of as a single-point 3D proxy because it consists of 3D markers at joint locations. It would be time-consuming to generate a 3D proxy by their method for a full animation because of the nature of the user-input involved.

Wei and colleagues [126] generate physically realistic human motion from uncalibrated monocular data. Their method relies heavily on the assumption that the human skeleton is comprised of rigid bones and that there are various symmetries in the human body. Newtonian physics, friction and contact forces inform other terms in their optimization, but need not be valid priors for traditional hand-drawn animation.

In addition to the work on 3D interpretations of a given hand-drawing, there has also been research on how a 2.5D popup may be used. Sykora and colleagues employ user-specified depth inequalities to generate a 2.5D popup that may be used for layering, shading and generating stereoscopic views [115]. Rivers and colleagues interpolate different views based on an underlying 2.5D model, so that a hand-drawn shape can be seen from a novel viewpoint [96].

Computer graphics techniques have also been used to create background scenery, either in the form of 2D paintings manipulated to look three dimensional [98, 127], or as a 3D scene, as in *Tarzan*'s Deep Canvas [22]. These approaches do not allow for physical interaction between the hand-drawn elements and the 3D elements—the CG background can be composited with the hand-drawn foreground, but does not interact with it, for example, there are no dynamically simulated ripples when *Tarzan* steps into a pool of water. Our work addresses the challenge of connecting a traditionally animated character with 3D CG elements by enabling the character to drive the motion of the 3D scene elements via its 3D proxy.

2.2.2 Computer Vision

The recovery of 3D human pose from images has been studied in the computer vision community for over three decades (see, for example, [76] and [37]). At a high level, these methods look for a 3D pose which minimizes the geometric projection error, and is constrained by priors about the way humans are proportioned and how humans move—these priors include limits on joint angles [44, 111], physical models of the body [100], foot plants as a constraint [101], and known limb lengths [63, 119]. Sidenbladh and colleagues [110] and Rosenhahn [99] further applied smoothness constraints across a video sequence. Articulation constraints, to ensure that limbs must remain connected at joints, have also been used in a number of approaches [12, 25, 129]. Recently, dimensionality reduction methods, which rely on motion capture data to learn mappings, have become popular [41, 109, 121]. Along with these generative approaches, a number of discriminative approaches have also been proposed. These methods learn regression functions to link appearance features to 3D structure [2, 9, 33, 87, 112].

While researchers have demonstrated that a variety of priors are useful to resolve the forward-backward depth ambiguity, the most important assumption made in all these approaches is that the input data has a true and consistent 3D interpretation. In contrast, in our domain, characters are hand-drawn, rather than being recorded via physical cameras, and talented animators often purposely violate the constraints of a skeleton-based model.

2.3 Approach

The frames that are drawn by the artist contain the perspective view of the animated character as seen from a stationary or moving camera. As a result, there are two types of ambiguity in creating a 3D proxy—the depth ambiguity and the ambiguity created by the composition of the character motion and the camera motion. The depth ambiguity occurs because multiple 3D points can project to the same 2D point (Figure 2.2(a)). The composite motion ambiguity occurs because the hand-drawn frames do not contain sufficient information to disambiguate the motion of the camera from the motion of the character. Figure 2.2(b) illustrates the camera-character motion ambiguity. For the purpose of articulated pose reconstruction, Interpretation 1 and Interpretation 2 are equivalent. However, when placing the character in a 3D virtual world, choosing the correct interpretation is essential or the skirt will not have the correct dynamic motion.

In this section, we will discuss how we preprocess raw data, estimate the camera that the artist might have imagined, and resolve the depth ambiguity for each of the three levels of 3D proxies—the single-point proxy, the tapered cylinder model, and the hierarchical joint model.

2.3.1 User Input

Because automatically tracking body parts in hand-made drawings is noisy, we ask a user to provide annotation of the hand-drawn animation. We also ask the user to select a motion capture segment to aid in resolving the depth and camera-character motion ambiguity. The user also provides an initialization that is used to estimate the artist's camera.

Annotating the hand-drawings

We ask a user (who can be a lay person) to specify the skeleton of the hand-drawn character with N virtual markers and the approximate bounding box for every limb. This annotation is done for each frame of the input animation. The user also provides a segmentation of the different body parts by color coding the interior of the hand-drawn figure (Figure 2.3). These user inputs are designed to fit into the traditional 2D animation workflow [21, 54]—the virtual markers and

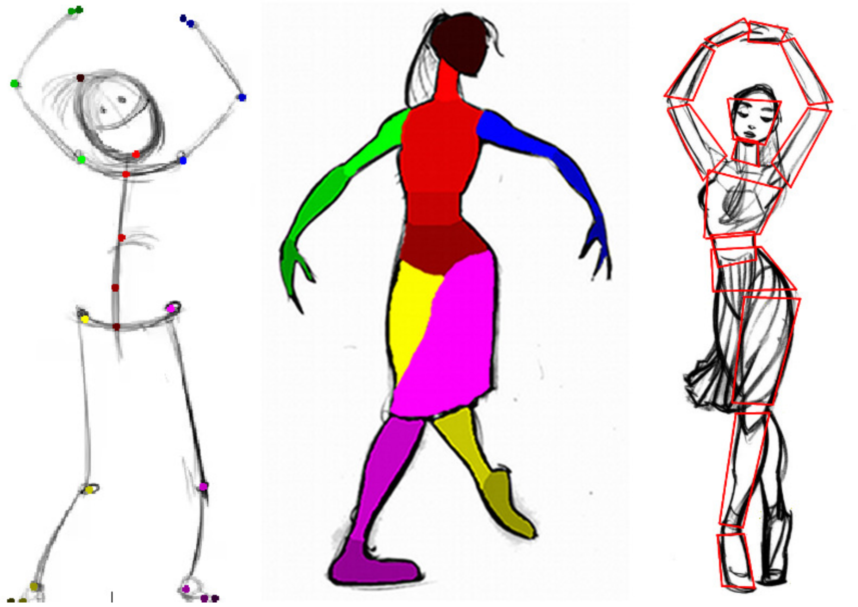


Figure 2.3: Annotation: The user annotates the drawing with markers for joint locations (left). Color segmentation of the different body parts for the fully-fleshed characters (middle). Approximate bounding boxes, shown in red, for the limbs (right). The bounding boxes for the right leg have been omitted for clarity.

bounding boxes can be marked when the cleanup or inbetweening artist re-touches every frame of the animated sequence, and the color segmentation can be done as part of the ink and paint process without requiring significant additional effort.

For each frame i, the user-specified virtual markers on the hand-drawn frames are denoted $\tilde{\mathbf{x}}_i = [\tilde{x}_1, \tilde{y}_1, \tilde{x}_2, \tilde{y}_2 ... \tilde{x}_N, \tilde{y}_N]^T$.

Selecting a motion capture segment

The user selects a motion capture segment that has a similar sequence of actions as the hand-drawn sequence, from the same point of view. The 3D poses in this motion capture segment will be used to resolve the depth ambiguity. The selection of the motion capture segment also helps resolve the composite camera-character motion ambiguity (Figure 2.2(b))—the system assumes that the root of the character moves according to the motion capture segment, and the remaining motion is camera motion. We, therefore, refer to this segment as a motion prior.

The motion prior can differ from the hand-animation in timing because we preprocess the segment via the Dynamic Time Warp algorithm [34, 105]. We denote the time-warped segment $\tilde{\mathbf{X}}$. For a frame i, the 3D marker positions for the motion prior poses are

$$\tilde{\mathbf{X}}_i = [\tilde{X}_1, \tilde{Y}_1, \tilde{Z}_1, 1, \tilde{X}_2, \tilde{Y}_2, \tilde{Z}_2, 1, ... \tilde{Z}_N, 1]^T, \tag{2.1}$$

expressed in homogeneous world coordinates. This sequence of poses could be a sequence produced by blending, interpolating, or editing poses from a database of motion capture or key frame motion.

Rotating a motion prior pose

As a first approximation to the camera imagined by the artist, we ask the user to specify an orthographic camera $\mathbf{R}_{2\times 3}$. Because roll can be reasonably assumed to be zero, this camera is characterized by two degrees of freedom—azimuth angle and elevation. The user specifies these angles through a graphical interface that allows the motion prior pose to be rotated until its orthographic projection visually matches the hand-drawing. The rotation angles are used to compute $\mathbf{R}_{2\times 3}$, an orthographic initialization for the artist's camera.

2.3.2 Preprocessing

When transferring the style of the hand-drawn animation onto a hierarchical 3D skeleton, there is a trade-off between tracking the artist's lines precisely and generating smooth, natural 3D motion. This trade-off exists because talented hand-animators purposefully violate the rigidity of the human skeleton to convey emotion via squash and stretch. A hierarchical joint skeleton does not provide the same affordances; thus, tracking a squashed arm accurately might result in unnatural movement for the elbow joint in the 3D proxy.

We introduce a pose descriptor to extract the pose of the hand-drawn character, while filtering out changes in limb length. This pose descriptor quantitatively describes a 2D hand-drawn pose, and is translation and scale invariant. The intuition here is that a character can be in the same pose at different locations, and two characters can have the same pose even if their relative limb lengths are different. In this preprocessing step, we modify the motion prior poses to match the pose descriptors for the hand-drawn poses.

For a given 2D pose, the descriptor starts at the root (which is the pelvis) and travels every hierarchical limb chain. For every link in the chain, we determine the position vector of the child marker in a coordinate frame fixed to its parent. As illustrated in Figure 2.4, the position vector for the wrist would be calculated with respect to a coordinate frame fixed to the elbow. The reference orientation for this coordinate frame can be absolute (i.e., oriented along the *x*-axis of the world coordinate frame), or relative (i.e., oriented along the corresponding limb, in this case, *right radius*). The pose descriptor $\mathbf{P}$ for a given pose would be the vector of polar angles for the position vectors of K virtual markers of the skeletal model,

$$\mathbf{P} = [\theta_1, \theta_2, ..., \theta_K]^T, \tag{2.2}$$

where K is the number of limbs that are needed to characterize the pose.

We first compute the pose descriptors for the hand drawings. Then the 3D motion prior poses $\tilde{\mathbf{X}}_i$ are projected to two dimensions with the camera approximation $\mathbf{R}$. The matrix $\mathbf{R}$ is block diagonal, where $\mathbf{R}_{2\times 3}$ is the repeated block. The projected motion prior poses are shown in Figure 2.4 (far left), where

$$\tilde{\mathbf{x}}_{2D_i} = \mathbf{R}\tilde{\mathbf{X}}_i.$$

The character model is divided into 'upper body', which consists of the hierarchical chains containing the two arms and the head, and 'lower body', which consists of the limb chains involving the legs (Figure 2.4). We start modifying the projected motion prior pose at the pelvis and work out along each hierarchical chain of the upper body. Each limb segment of the projected

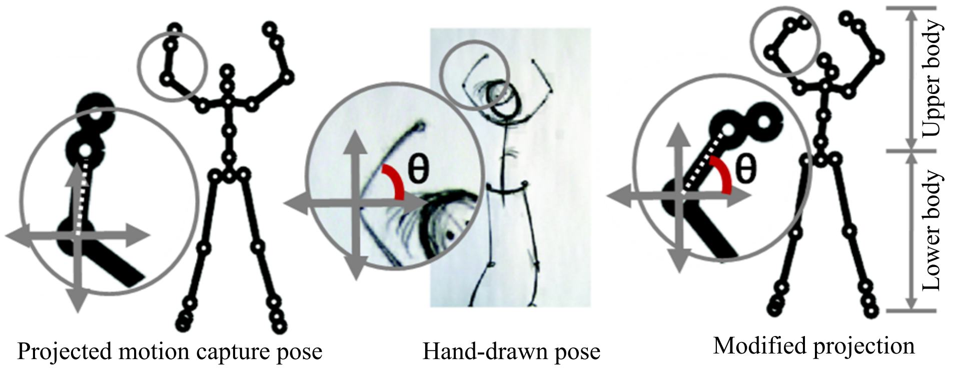


Figure 2.4: Preprocessing: The pose descriptor consists of in-the-image-plane angles for every limb segment. The limb segments of the projected motion prior pose are modified to match the pose descriptor for the hand drawn pose via planar rotation.

motion prior pose is rotated in the image plane so that the in-plane polar angle is the same as the desired pose descriptor, that is, the corresponding polar angle in the hand-drawn pose (Figure 2.4 insets).

The modified 2D motion prior pose, $\tilde{\mathbf{x}}_i^m$, is illustrated in Figure 2.4 (far right). The lower body is not modified in order to transfer the ground contacts of the motion capture data onto the 3D joint hierarchy.

2.3.3 Camera Estimation

As illustrated in Figure 2.2(b), it is essential to resolve the camera-character motion ambiguity to place a 3D proxy in a virtual 3D world. However, the hand-drawings alone do not contain sufficient information to disambiguate the motion of the camera from the motion of the character. We resolve this ambiguity by registering the poses of the time-warped motion prior $\tilde{\mathbf{X}}_i$ with the hand-drawn poses $\tilde{\mathbf{x}}_i$. The underlying assumption is that the root of the character moves according to the motion prior, any movement in the hand-drawn markers over and above the movement of the motion prior poses is attributed to the camera.

For each frame i, we estimate a projection matrix $\mathbf{M}_i$ that minimizes the geometric projection error, e_g,

$$e_g = \sum_{t=-K/2}^{K/2} ||\tilde{\mathbf{x}}_{i+t} - \mathbf{x}_{i+t}^{proj}||,$$

where $\mathbf{x}_{i+t}^{proj} \cong \mathbf{M}_i \tilde{\mathbf{X}}_{i+t}$. We compute across a moving window of K frames around the frame i to increase robustness to noise.

In addition to minimizing projection error, we also want $\mathbf{M}_i$ to characterize a physically realizable camera that can render 3D elements: skew and tilt are set to zero, the scale factors are computed from the image resolution, and the focal length is pre-specified. This formulation is similar to Hornung and colleagues [46] and Petrovic and colleagues [83]. As a result, only

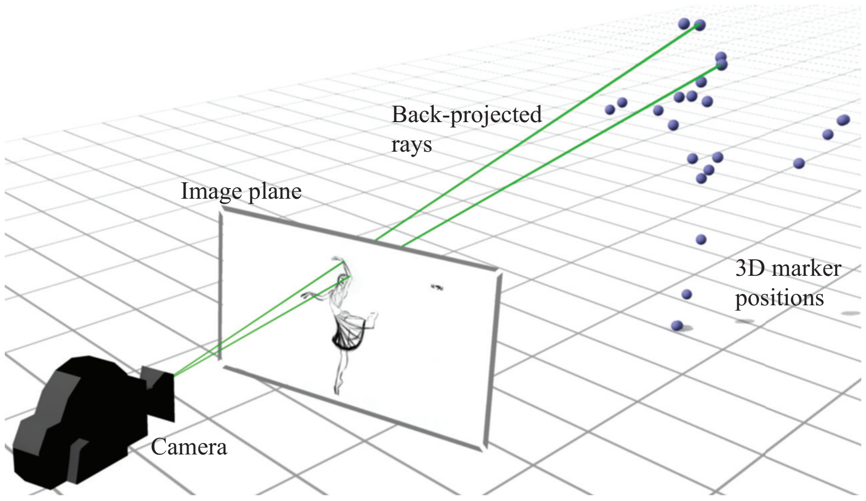


Figure 2.5: Attachment points: Markers on the hand drawing are back projected. Depth is obtained from the motion prior. The blue 3D markers illustrate what the solution looks like for a chosen frame.

the external parameters for $\mathbf{M}_i$ remain to be estimated: roll, pitch, yaw, and the location of the center. We denote the external parameters $\rho(i) = (\theta_x(i), \theta_y(i), \theta_z(i), t_x(i), t_y(i), t_z(i))^T$.

Constraints on the external parameters are that the renderable camera should be above ground level, $e_l = (t_z - \mu)$, roll should be minimum, $e_r = |\theta_y|$, and the camera should move smoothly, $e_s = ||\rho(i) - \rho(i-1)||$. Therefore, we estimate $\rho^*(i)$ such that

$$\rho^*(i) = \underset{\rho}{\operatorname{argmin}}(\omega_1 e_g + \omega_2 e_l + \omega_3 e_r + \omega_4 e_s), \tag{2.3}$$

where $\omega_1, \omega_2, \omega_3$ and ω_4 are the associated weights. In practice, $\omega_1 = 1, \omega_2 = \omega_3 = \omega_4 = 0.0005$ and $\mu = 1$. We use a nonlinear optimizer, with the roll initialized to zero, the yaw and pitch set to the user-provided orthographic initialization, and the location of the center initialized approximately so that the first motion capture pose fits inside the image plane.

2.3.4 Detail-dependent Minimization for 3D Proxy

Figure 2.1 illustrates the levels of 3D proxies for a hand-drawn character. We generate every 3D proxy by minimizing a weighted sum of three terms. Depending on the function of the 3D proxy, the individual terms are reformulated to best reflect the properties desired in the solution.

The first term is called the *input-matching* term e_a and causes the 3D proxy to follow the hand-animation. Because the artist provides only the perspective drawing of the character, we need to infer the missing depth information. The second error term is the *motion prior*, e_m, which provides a data-driven inference for depth. The third term is the *smoothing term*, e_s. These terms can be related to the formulation of 3D pose reconstruction in the computer vision literature: the input-match term, e_a, is analogous to the geometric projection error, the motion prior, e_m, is analogous to the data-driven or physics priors, and the smoothing term, e_s, is analogous to temporal regularization.

Attachment Points

To attach a 3D element to a hand-drawn character (for example, a simulated balloon onto the wrist of a hand-drawn figure), the level of detail for the proxy is plausible 3D trajectories for the attachment points. These 3D trajectories must achieve perfect image plane alignment with the hand-drawn character—only then will the 3D element appear to be convincingly attached to the artist-drawn character.

For each frame i, $\mathbf{M}_i$ is the projection operator and the 3D position of each marker j is denoted $\mathbf{X}_{ij}^{w} = [X_{ij}^{w}, Y_{ij}^{w}, Z_{ij}^{w}, 1]^T$ in homogeneous world coordinates. Then, perfect image plane alignment is achieved by minimizing the input-match error e_a,

$$e_a = ||\tilde{\mathbf{x}}_{ij} - \mathbf{x}_{ij}^{proj}||, \tag{2.4}$$

where $\tilde{\mathbf{x}}_{ij}$ are the virtual markers on the hand-drawing, and $\mathbf{x}_{ij}^{proj}$ are the projections of the corresponding markers of the 3D proxy,

$$\mathbf{x}_{ij}^{proj} \cong \mathbf{M}_i \tilde{\mathbf{X}}_{ij}^{w}.$$

There can be infinitely many solutions which minimize the input-match error in Equation 2.4. The motion prior term e_m resolves this ambiguity by pulling the z-depth for each marker as close as possible to the corresponding value for the motion prior poses, $\tilde{\mathbf{X}}$. For the i^{th} frame,

$$e_m = ||m_3^T \tilde{\mathbf{X}}_{ij} - m_3^T \mathbf{X}_{ij}^{w}|| \qquad \forall j = 1, ..., N. \tag{2.5}$$

The smoothing term e_s keeps the final trajectory temporally coherent. For each marker j,

$$e_s = ||\mathbf{X}_{ij}^{w} - \mathbf{X}_{(i+1)j}^{w}||. \tag{2.6}$$

Finally, normalization constraints fix the scale factor in homogeneous coordinates to unity.

$$[0, 0, 0, 1]\,\mathbf{X}_{ij}^{w} = 1 \qquad \forall j = 1, ..., N. \tag{2.7}$$

The Appendix contains details for how these terms are composed into a linear system, which can be solved in closed form. Figure 2.5 illustrates the solution for one frame geometrically.

Collision Volumes

To create believable interaction between the hand-drawn character and 3D objects (for example, splashes when the hand-drawn figure collides with simulated water, or a simulated scarf wrapping around the waist of the hand-drawn figure), the proxy consists of 3D polygonal shapes that track the hand-drawn lines. Because the 3D proxy only provides collision volumes and is never directly rendered, the polygonal shapes need not conform exactly to the hand-drawn lines. The three error terms, e_a, e_m and e_s, are reformulated to reflect these new constraints. The polygonal shapes used here are tapered cylinders for the arms, legs and torso, and spheres for the joints, but other shapes would be feasible too.

The 3D polygonal shapes are generated by first computing the 3D positions of the end points of the cylinders. Here, as we explain the details for one limb, we will drop indices for clarity.

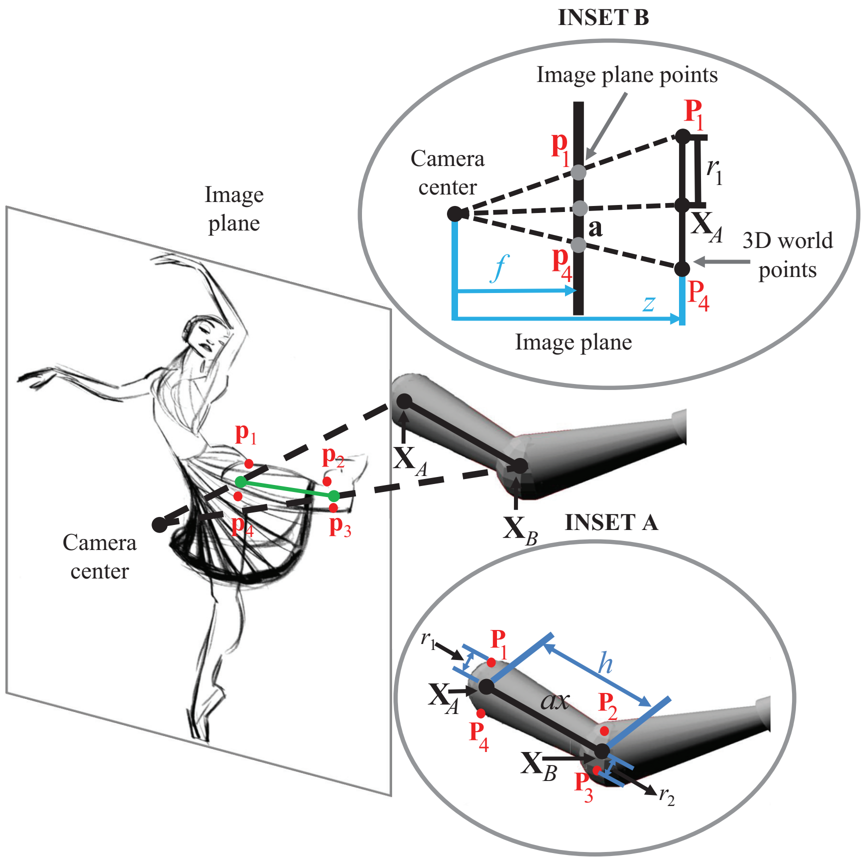


Figure 2.6: Collision volumes: User-specified markers are back projected to obtain the 3D marker positions $\mathbf{X}_A$ and $\mathbf{X}_B$. **Inset A**: The cylindrical collision volume is characterized by its axis and height, and the radii of either face. **Inset B**: This inset describes how we compute the radius of one face of the cylinder. The image plane points $\mathbf{p}_1, \mathbf{p}_4$ are back projected to $\mathbf{P}_1, \mathbf{P}_4$ such that the z-depth is the same as the z-depth for the marker A.

Figure 2.6 shows the 3D positions, $\mathbf{X}_A$ and $\mathbf{X}_B$, of the upper leg markers. This computation is identical to Section 2.3.4.

A tapered cylinder collision volume is completely characterized by the direction of its cylindrical axis, its height, and the radii of either face. The direction of the cylindrical axis $\vec{ax}$ and the height of the cylinder h are determined from the end point positions, $\mathbf{X}_A$ and $\mathbf{X}_B$ (Figure 2.6, Inset A). The radii of the two faces, r_1 and r_2, depend on the thickness of the limb, as drawn by the artist. A simple algorithm approximates the artist-drawn limb with a bounding box, shown in Figure 2.6 as the quadrilateral $\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_4$. This algorithm looks for the artist-sketched line by locating the first black pixel in the direction perpendicular to the line joining markers A and B. Overlap between body parts will confuse this algorithm. When the arm crosses the torso for example, the algorithm incorrectly marks the boundary of the arm as the torso boundary. These cases are corrected by the user.

Intuitively, the radii of the two faces, r_1 and r_2, are determined by back projecting the image plane bounding box $(\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_4)$ to the same z-depth as the 3D markers A and B (Figure

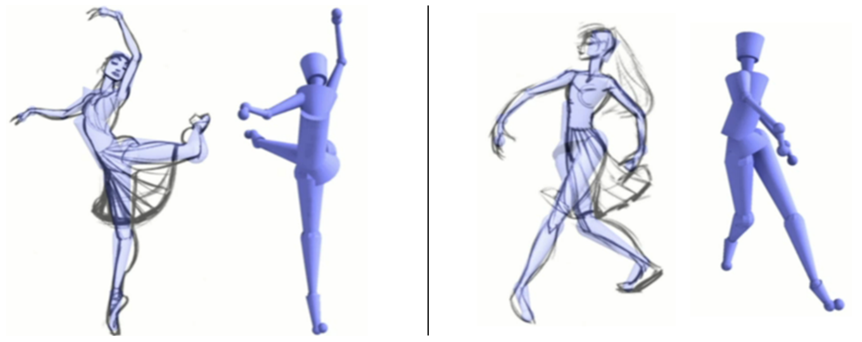


Figure 2.7: Collision volumes: The hand-drawings are shown overlaid with the collision volumes. The collision volumes match the drawings approximately, but need not exactly conform to the artist's lines. We also show the collision volumes from an alternate viewpoint.

2.6, Inset B). Then, we spin the back projected quadrilateral about the axis AB and the volume of revolution is the tapered cylinder proxy for the limb AB. Formally, for each point $\mathbf{p}_q (q = 1, 2, 3, 4)$, let $\mathbf{P}_q$ denote their back projected 3D positions. The input-match term e_a is redefined so that the 3D positions $\mathbf{P}_q$ align with the image plane bounding box,

$$e_a = ||\mathbf{p}_q - \mathbf{p}_q^{proj}||, \tag{2.8}$$

where $\mathbf{p}_q^{proj} \cong \mathbf{M}\mathbf{P}_q$. The error term e_m is redefined so that the z-depth for $\mathbf{P}_q$ is the same as the z-depth for the markers A and B (illustrated in Figure 2.6, Inset B),

$$e_m = ||m_3^T \mathbf{P}_q - m_3^T \mathbf{X}_A|| \quad \text{for } q = 1 \text{ and } 4, \tag{2.9}$$
$$e_m = ||m_3^T \mathbf{P}_q - m_3^T \mathbf{X}_B|| \quad \text{for } q = 2 \text{ and } 3. \tag{2.10}$$

We do not have a smoothing term across frames because the 3D positions $\mathbf{X}_A$ and $\mathbf{X}_B$ are already temporally smooth. The weighted sum of the error terms is linearized and minimized as detailed in the Appendix. The radii for the faces of the tapered cylinder are computed,

$$r_1 = \frac{\sqrt{||\mathbf{P}_1 - \mathbf{P}_4||}}{2}, \; r_2 = \frac{\sqrt{||\mathbf{P}_2 - \mathbf{P}_3||}}{2}. \tag{2.11}$$

The sphere for a joint has the same radius as the corresponding tapered cylinder for the limb. Figure 2.7 shows the collision volumes overlaid with the hand-drawings on the left, and an alternate viewpoint of the same pose on the right of each part. The 3D proxy does not exactly conform to the artist's lines, but is sufficient to create believable interactions with dynamic simulations like cloth. Because the collision volumes can change size and shape independent of each other, this 3D proxy is well-suited to track the body of a hand-animated character, even when the artist changes its size and shape (for example, if a dancer is drawn taller in frame 10 relative to frame 1 to create the illusion of reaching up).

Skinned, Lit, and Rendered Model

The 3D proxy for the hand-drawn character may be a skeletal model, which is controlled via a joint hierarchy, and can be skinned, lit and rendered. When creating this proxy, the goal is to create 3D motion for the skeleton based on the artist's hand-drawn animation. Because this 3D proxy can be rendered from any possible viewpoint, the error terms in our minimization are reformulated to maintain naturalness of motion and preserve ground contacts while transferring the style of the hand animation onto the skeleton.

The user-specified virtual markers on the hand drawings, $\tilde{\mathbf{x}}_{ij}$, contain the squash and stretch used by the artist to convey emotion. Because there is no precise mathematical model for how artists extend and compress the limbs of a character, the intentional change in limb lengths is indistinguishable from imprecision in the drawing and noise introduced during annotation. However, the limb lengths for the skeletal model are fixed. As a result, computing joint angle values directly from the user-specified virtual markers leads to large errors in joint angles.

We reformulate the input-match error term to filter out some of this noise. Recall that for each hand-drawn pose $\tilde{\mathbf{x}}_i$, the corresponding pose in the motion prior segment, $\tilde{\mathbf{X}}_i$, was preprocessed so that the pose descriptors matched (Section 2.3.2). The upper body of the motion prior pose was modified while the lower body was left unchanged to preserve ground contacts. The input-match term for the skeletal model proxy aligns the 3D proxy to the modified pose $\mathbf{x}_{ij}^m$, instead of the original hand-drawn pose $\tilde{\mathbf{x}}_i$. Thus,

$$e_a = \|\mathbf{R}\mathbf{X}_{ij}^w - \mathbf{x}_{ij}^m\|. \tag{2.12}$$

The motion prior pose $\tilde{\mathbf{X}}_i$ is a reasonable prior for the desired 3D marker positions $\mathbf{X}_i^w$. Thus, the motion prior error term, e_m, is simply

$$e_m = \|\mathbf{X}_i^w - \tilde{\mathbf{X}}_i\|. \tag{2.13}$$

Because the individual limbs of the 3D skeletal model are not squashable or stretchable, we additionally enforce constant limb lengths. Let the 3D positions of the two virtual markers on the ends of a given limb be $(x_{j_1}, y_{j_1}, z_{j_1})$ and $(x_{j_2}, y_{j_2}, z_{j_2})$. Then, the computed squared length of the limb, l_j^2, is

$$l_j^2 = (x_{j_1} - x_{j_2})^2 + (y_{j_1} - y_{j_2})^2 + (z_{j_1} - z_{j_2})^2. \tag{2.14}$$

This computed length must be equal to the actual skeletal length L of the same limb, computed from the motion prior data. Mathematically,

$$e_s = \|l_j^2 - L^2\|. \tag{2.15}$$

We can linearize Equation 2.15 using a Taylor series expansion around the corresponding motion prior pose and stack the length equations for each limb to yield

$$e_s = \|\mathbf{A}_{limb}\mathbf{X}_i^w - \mathbf{b}_{limb}\|. \tag{2.16}$$

where $\mathbf{A}_{limb}$ and $\mathbf{b}_{limb}$ are functions of the motion prior pose $\tilde{\mathbf{X}}_i$ and L.

The three error terms are stacked together as a linear system of equations:

$$\begin{aligned} \mathbf{W}\begin{bmatrix} \mathbf{R} \\ \mathbf{A}_{limb} \\ \mathbf{I} \end{bmatrix}\mathbf{X}_i^w &= \begin{bmatrix} \tilde{\mathbf{x}}_i^m \\ \mathbf{b}_{limb} \\ \tilde{\mathbf{X}}_{ij} \end{bmatrix}, & (2.17) \\ \mathbf{W}\mathbf{A}_{full}\mathbf{X}_i^w &= \mathbf{b}_{full}. & (2.18) \end{aligned}$$

where $\mathbf{W}$ contains the weights of the various error terms.

Typically, the number of markers for each pose is $N = 24$, which makes $\mathbf{X}_i^w$ a vector of length 72. The work of Safonova and colleagues [104] shows that most dynamic human motions can be described by a low-dimensional PCA (Principal Components Analysis) subspace. Building on this result, we look for the solution to Equation 2.18 in a low-dimensional subspace of an activity specific motion capture database. Let μ denote the mean of the motion data and v_i denote the basis vectors or the principal components obtained through PCA. Further, let x_i^b be the coordinates in PCA space, and $\mathbf{V}$ be a $3N \times P$ matrix of basis vectors.

$$\begin{aligned} \mathbf{X}_i^w &= \mu + \sum_{i=1}^{P} x_i^b v_i && (2.19) \\ &= \mu + \mathbf{V}\mathbf{X}_i^b, && (2.20) \end{aligned}$$

where $\mathbf{X}_i^b$ is a vector comprising the coordinates for each of the P basis vectors. The weighted least squares system in Equation 2.18 can then be written as

$$\begin{aligned} \mathbf{W}\mathbf{A}_{full}\left(\mathbf{V}\mathbf{X}_i^b + \mu\right) &= \mathbf{b}_{full}, && (2.21) \\ \mathbf{W}\mathbf{A}_{full}\mathbf{V}\mathbf{X}_i^b &= \mathbf{b}_{full} - \mathbf{A}_{full}\mu, && (2.22) \\ \mathbf{W}\mathbf{A}\mathbf{X}_i^b &= \mathbf{b}. && (2.23) \end{aligned}$$

We can find the least squares solution to Equation 2.23 and reproject $\mathbf{X}_i^b$ to get the 3D marker positions $\mathbf{X}_{ij}^w$. As this system is linear, the solution is the global minimum, is numerically stable, and can be found in closed form.

In a hierarchical skeletal model, every limb is described by three joint angles (roll, pitch and yaw) relative to its parent limb. We convert the marker positions $\mathbf{X}_i^w$ into joint angles. The root joint for our character model is the pelvis, therefore, we start by recovering the rotation of the pelvis with respect to the world coordinate frame and work our way through each hierarchical chain to generate the full pose [50]. The joint angles for our skeletal model describe the rotation of the limb segment in the xyz ordering—when we convert this description to the zyx ordering, θ_x and θ_y are functions of 3D marker positions (roll and pitch), and θ_z is the 'yaw' angle, which cannot be computed from marker positions. We find the rotation of the corresponding limb segment in the motion prior, and simply use that θ_z to complete the generated 3D pose (Figure 2.8).

2.3.5 Interaction between Proxy and Scene Elements

The 3D proxy model can be imported into any commercially available modeling and animation software package. As a 3D scene element, the proxy can interact with other objects in the scene. For example, the hierarchical joint model can be imported into a Maya scene and skinned. We can put a dynamically simulated skirt on the 3D model, relight the scene and move the camera around, as in Figure 2.17.

In Figure 2.9, the collision volumes for the jumping jacks character have been imported into a Maya scene. An artist has created pompoms and a skirt. The Maya dynamics engine [113] is used to physically simulate the motion of the pompoms and the skirt, and their interaction with

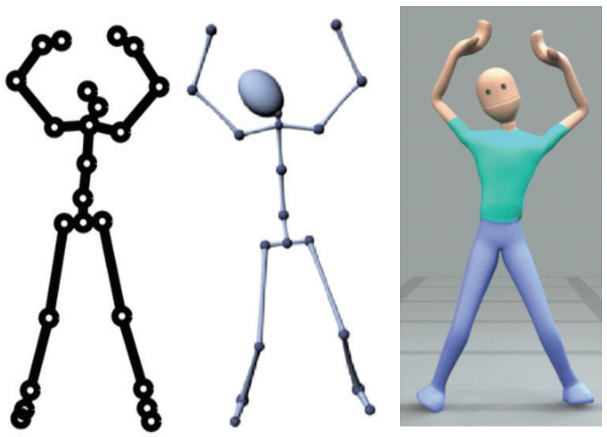


Figure 2.8: Modified motion capture pose, 3D markers, hierarchical skeletal model.

the 3D collision volumes of the hand-drawn character. Maya is also used to render the 'beauty' pass, Υ_i^r, which contains the scene elements with texture and lighting, the depth map Δ_i^r, and the occlusion map η_i^r (Figure 2.9(c-e)).

We composite the rendered skirt and pompoms with the hand-animation to obtain simulated secondary motion for the jumping jacks character (the same procedure is used for all the animation sequences). The depth map Δ_i^h for the hand-drawn image is computed by linearly interpolating known depths. For the skinned characters, the pixels belonging to a given limb are obtained by color segmentation (color-coding done as part of user input in Section 2.3.1). For stick figures, we segment out the dark pixels by thresholding inside an oriented window along the limb v ($v = 1, 2, ..V$).

The z-depth values for the pixels corresponding to the N virtual markers are known and can be interpolated to generate Δ_i^h. For simplicity, we will drop the indices and denote the known depths as $\tilde{\mathbf{x}}$. Let $\mathbf{l}$ denote the line joining the end-point markers for limb v, whose image positions are $\tilde{\mathbf{x}}_a = (a_x, a_y)$ and $\tilde{\mathbf{x}}_b = (b_x, b_y)$. Then, l can be computed

$$\mathbf{l} = \frac{\tilde{\mathbf{x}}_b - \tilde{\mathbf{x}}_a}{||\tilde{\mathbf{x}}_b - \tilde{\mathbf{x}}_a||}. \tag{2.24}$$

Every pixel $\tilde{\mathbf{p}} = (\tilde{p}_x, \tilde{p}_y)$ belonging to the limb is assigned the same depth as the point $\mathbf{p}$ closest to it on $\mathbf{l}$. We perform this interpolation for every limb in turn to obtain the depth Δ_i^h, and then scale it to match the units of Δ_i^r (Figure 2.10).

The occlusion map η_i^h for the hand-drawn frame is

$$\eta_i^h = \begin{cases} 1, & \text{in the interior of the hand-drawn figure,} \\ 1, & \text{on the artist's lines,} \\ 0, & \text{otherwise.} \end{cases} \tag{2.25}$$

The alpha matte α for the hand-drawn frame Υ_i^h is defined as the inverse of the gray-scale value,

$$\alpha = (255 - \Upsilon_i^h)/255. \tag{2.26}$$

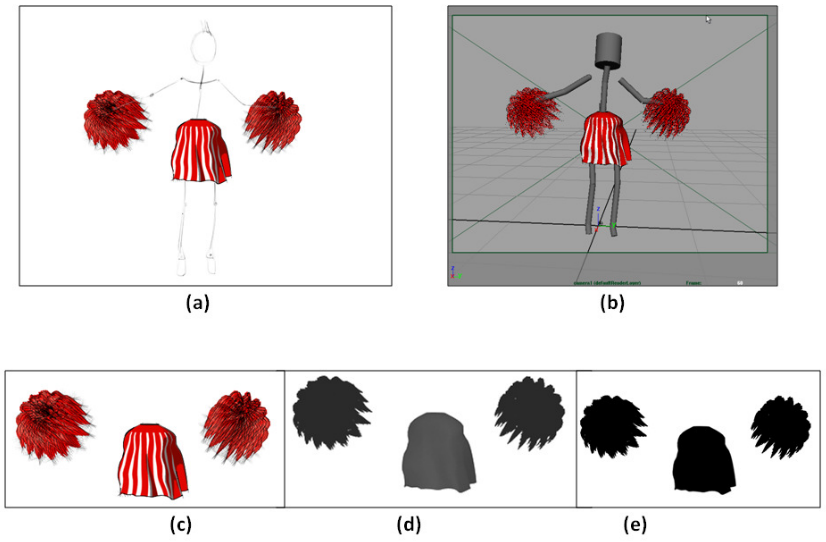


Figure 2.9: (a) The final composited frame with the hand-drawn character and the rendered 3D elements. (b) Collision volumes imported into the 3D animation package. (c) rendered scene elements. (d) z-depth for the rendered elements. (e) occlusion map for the rendered elements.

This alpha matte is modified to incorporate depth ordering. For all pixels p where $\eta_i^h(p) = 1$,

$$\alpha_{\text{new}} = \begin{cases} \alpha & \text{if} \quad \Delta_i^h < \Delta_i^r, \\ 0 & \text{otherwise.} \end{cases} \tag{2.27}$$

The final composited image I_{final} is computed

$$I_{\text{final}i} = \alpha_{\text{new}} \Upsilon_i^h + (1 - \alpha_{\text{new}}) \Upsilon_i^r. \tag{2.28}$$

2.4 Results

We present results on a variety of hand animated characters—a ballet dancer, a goofy character doing jumping jacks, a character doing a stylized walk across the screen, a ‘happy flower’, a little girl twirling, and a character ducking low. The hand animations include stick figures and fully fleshed characters, as well as stationary cameras and tracking cameras.

Figure 2.11 shows sample frames from the animated sequence of a character performing a stylized walk. The example in the top row shows the deformation of the snow, driven by a 3D cylinder proxy that approximates the character’s foot. In the bottom row, the umbrella is attached to the 3D marker for the wrist of the character. Simulated rain splashes and bounces off the 3D umbrella. In Figure 2.12, the hand-drawn character ‘holds’ a simulated 3D balloon and ‘kicks’ the balls on the ground. The 3D balloon is attached to the single-point 3D proxy for the hand-drawn character, that is, the 3D trajectory of the wrist marker. Cylinder proxies for the feet cause the balls on the floor to move.

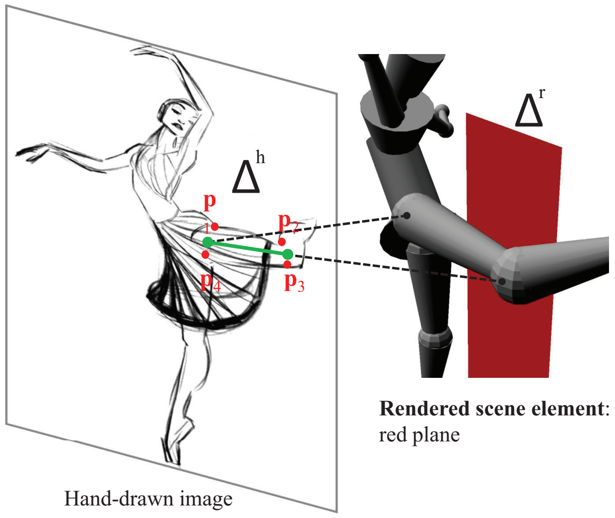


Figure 2.10: Depth ordering: Our method generates an alpha map for the hand-drawn image that maintains depth ordering between the hand-drawn pixels and the rendered 3D scene elements. The depth map for the 3D elements are rendered. The depth map for the hand-drawn character is computed from known marker depth values.

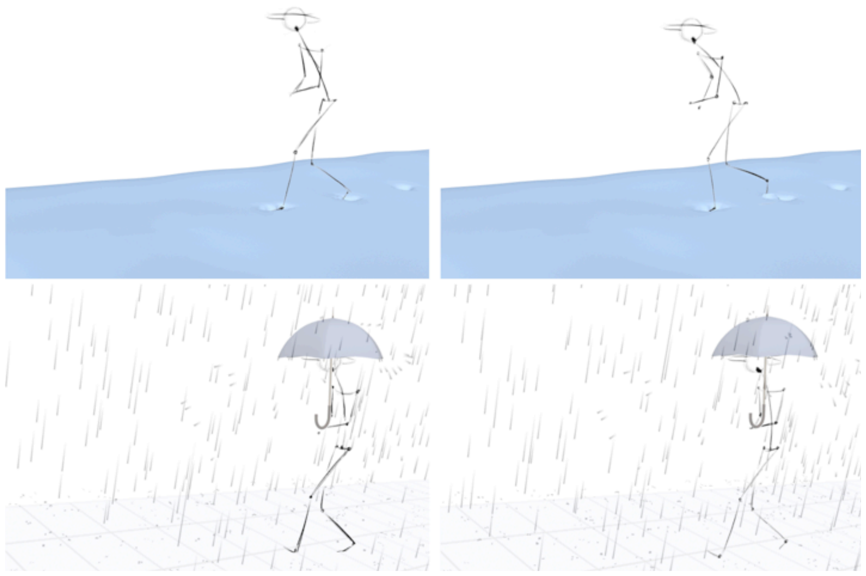


Figure 2.11: Stylized walk across the screen: In the top row, snow deforms as the character steps through it. The deformation is generated by cylinder proxies for the feet of the character. In the bottom row, simulated rain bounces off a 3D umbrella attached to the wrist. The umbrella is attached to the hand-drawn character via a single-point proxy for the wrist.

Figures 2.13 and 2.14 show results with cloth simulations. In Figure 2.13, the 3D proxy for the little girl is the tapered cylinder model. The skirt is a cloth simulation driven by the 3D proxy, then rendered and composited onto the hand animation. Thus, the simulated skirt appears to be

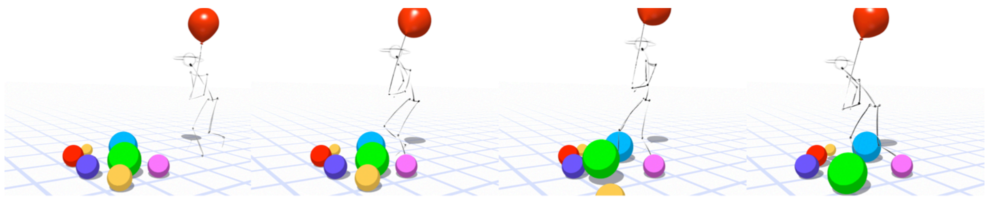


Figure 2.12: Stylized walk across the screen: The dynamics of the balloon, its string, and the colored balls are driven by the motion of the hand-drawn character.

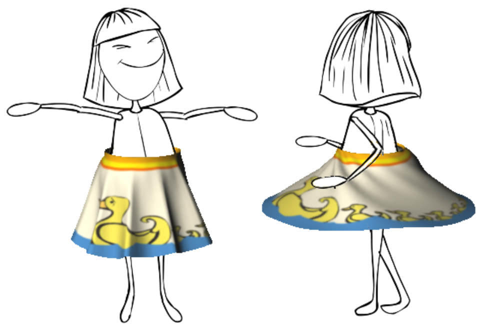


Figure 2.13: Twirling girl: Two frames from an animation of a little girl showing off her new skirt. The girl is hand-animated. The skirt is a 3D cloth simulation driven by a 3D cylinder proxy for the girl's body.

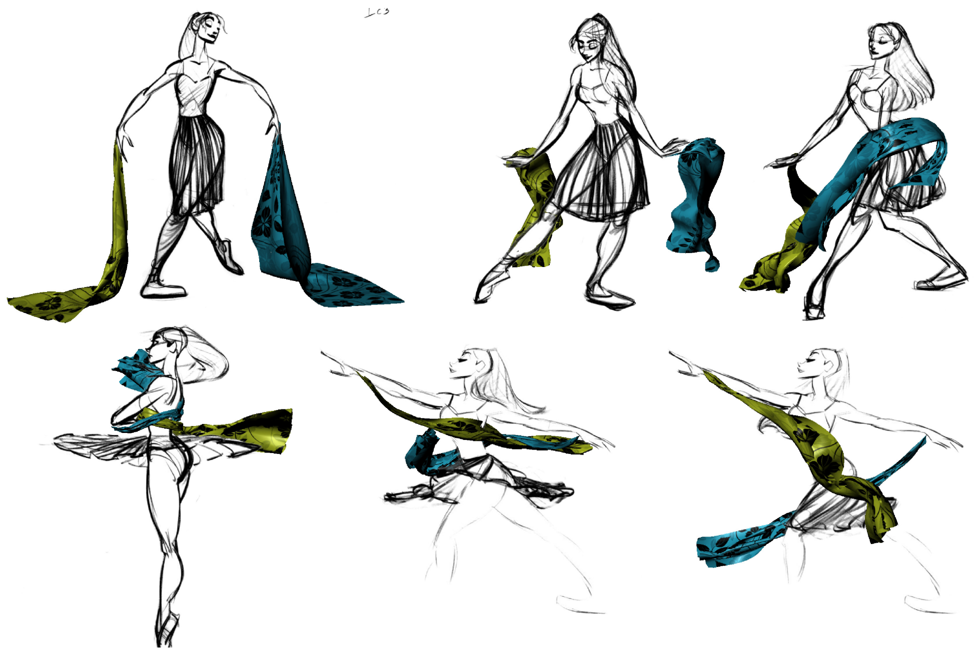


Figure 2.14: Ballet dancer: Scarves are simulated as 3D cloth. They are attached to the wrists of the dancer via single-point proxies for the wrist, and interact with the 3D cylinder proxy for the body of the hand-drawn ballerina, and thus, wrap around her waist.

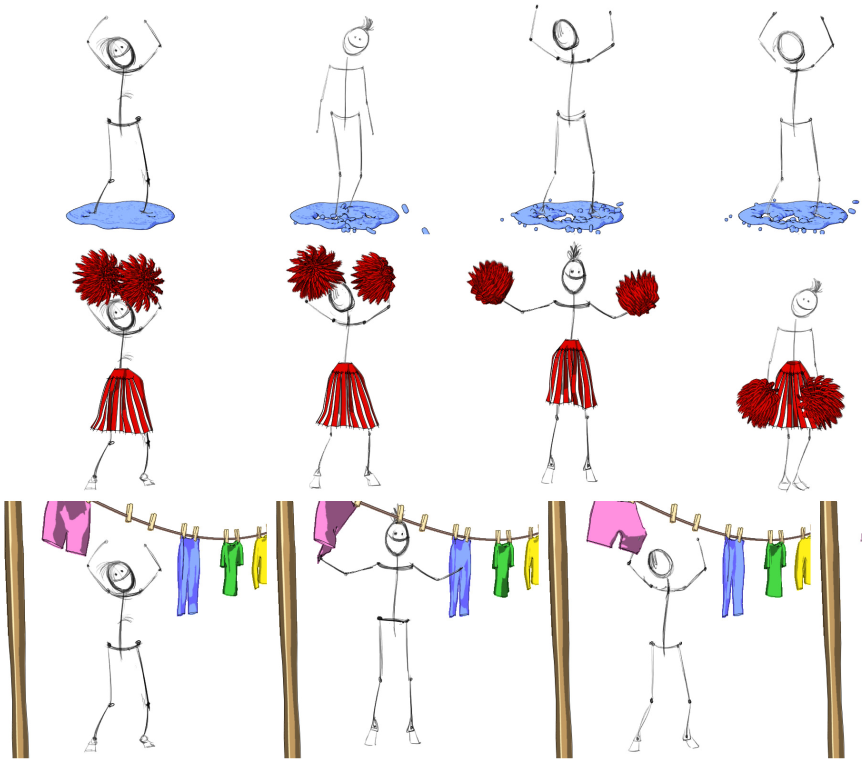


Figure 2.15: Goofy character doing jumping jacks: Water splashes in response to the feet (top row); the pom-poms deform in a physically realistic way, and the skirt billows around the character's legs (middle row); and the character's hand interacts with clothes on the clothesline (bottom row).

twirled by the hand-drawn little girl. In Figure 2.14, two scarves are attached to the wrists of the dancer. The cloth simulation is driven by the three-dimensional trajectory of the wrist markers, and interacts with the 3D polygons for the body of the ballerina.

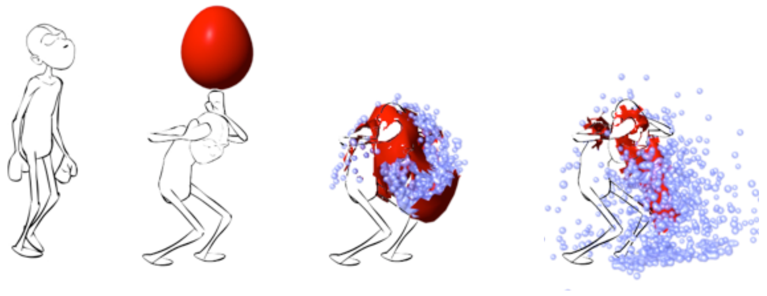


Figure 2.16: Pedestrian: A simulated water balloon falls from the sky on a hand-drawn pedestrian. The balloon is a cloth simulation and is filled with water particles. Both the cloth and the water interact with the collision volumes for the hand-drawn character. The camera tracks the character as he walks and stops when he ducks.

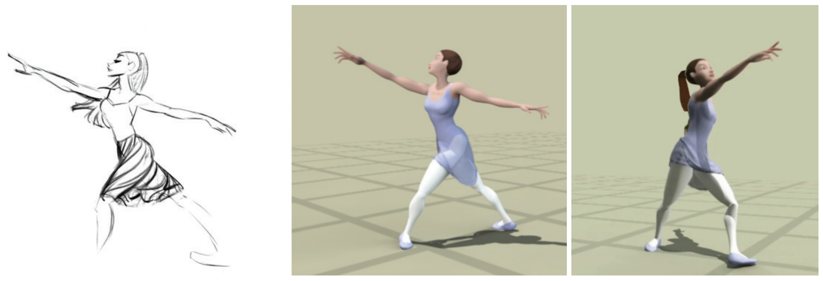


Figure 2.17: The hand-drawn frame with the corresponding renderings from two views. The skirt is a dynamically simulated element. The lighting can be changed easily.

Figure 2.15 demonstrates the advantage of leveraging 3D simulation to create secondary motion for a hand animation. Delicate effects like the strands of the pompoms, water splashing in a realistic way, and the bounce of a clothesline would all be time-consuming to hand animate to a comparable degree of detail. The pompoms are attached to the character's wrist, while all the other effects use the tapered cylinder 3D proxy for the hand-drawn character. In Figure 2.16, a water balloon falls from the sky on a pedestrian. The pedestrian is a hand-animated character. The balloon is a simulated cloth object and the water inside it is comprised of particles. The balloon bursts when it comes in contact with the 3D proxy for the character and we see it snag on his body on subsequent frames.

We also present examples where we have transferred the style of the hand-animation onto hierarchical joint models. In Figure 2.17, we see the 3D ballerina from two different camera viewpoints. The joint model proxies for the stylized walk, jumping jacks and 'happy flower' sequences are compared with the corresponding motion prior poses in Figure 2.18—note how the arms in the 3D proxy pose match the hand-drawn character, even though the motion prior pose is different.

In all the results presented, user effort can be divided into three parts:

- Cleanup/ink-and-paint stage: Marking out dots or user-specified virtual markers (1 minute per frame), marking bounding boxes (3-4 minutes per frame), color segmentation of body parts (7-10 minutes per frame using a stylus and tablet, and a standard brush-based paint program)
- Selecting a motion capture segment: 20-30 minutes.
- Creating 3D simulation in Maya: 2-10 hours (2 hours for the simple rigid bodies, 10 hours for the cloth). The tuning time is dependent on the user's familiarity with the tool and is equivalent to the tuning required to add any visual effect to a 3D animation. Maya can be replaced by any other simulation engine.

2.5 Evaluation

In our method, the user provides the motion prior by selecting a suitable motion capture segment from a database. This prior influences different error terms in the minimization depending on the required 3D proxy. We examine the dependence of our method on the motion prior through

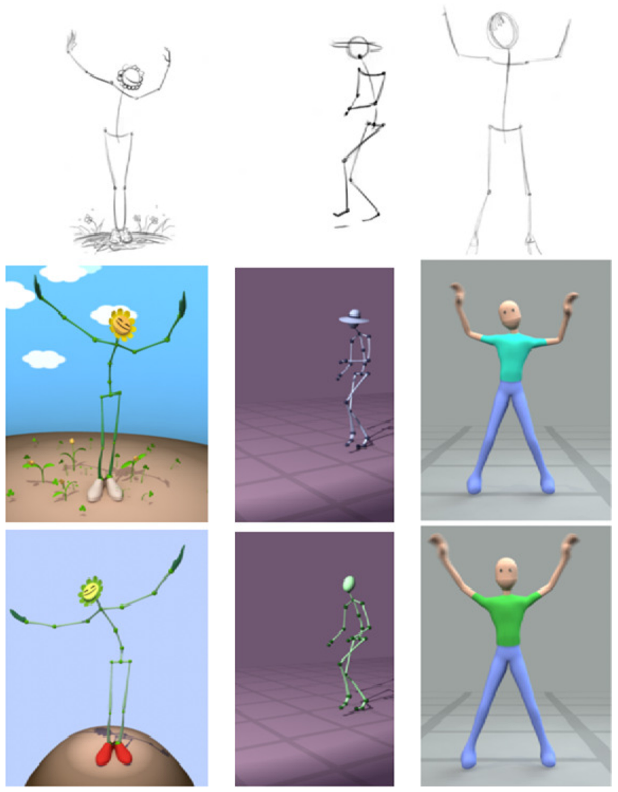


Figure 2.18: For each character, the top row is the hand-drawn animation, the middle row shows our result and the bottom row shows the corresponding motion capture poses. We found that the playful nature of jumping jacks animation (which was quite easily communicated by the artist on paper) was quite difficult for a human subject to perform because it is hard to perform a vigorous action like jumping jacks while bobbing the head in sync.

a synthetic example: a motion capture walk sequence (normal walk in a straight line) projected to 2D.

We examine the dependence of the 3D proxy on motion capture data when this data is only used to compute z-depth. We compute the error in the 3D positions of virtual markers when the 2D markers of the synthetic example are back projected under a known projection operator, and the z-depth is provided by different motion capture segments. Essentially, the motion prior $\tilde{\mathbf{X}}$ in Equation 2.5 is computed from five motion capture segments—a walk sequence from a different actor, a run, a broad jump, a side shuffle and a walk along a curve. Error is defined as the difference in z-depth value from ground truth, averaged over all N markers. Figure 2.19 illustrates that broad jump and run have similar numerical error in depth—the depth ordering for the limbs is similar for these actions when viewed in profile. Figure 2.20 shows sample frames when the z-depth is provided by a 'happy walk' motion capture segment. Even though the motion capture walk is stylistically different (arms swing more, legs step out further), the result matches the ground truth quite closely. Figure 2.21 shows sample frames when the z-depth is provided by a 'curved walk'. Because the actor curves towards the camera, the changing depth information causes our result to curve towards the camera as well.

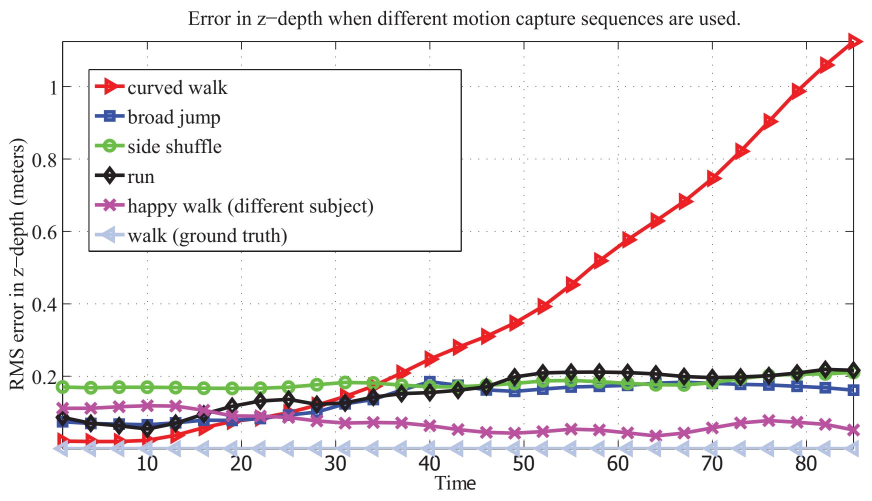


Figure 2.19: Different motion capture segments affect the error in z-depth. The normal walk is the ground truth. The z-depth error for the curved walk increases as the motion capture poses veer towards the camera. The lowest error is seen in a happy walk sequence captured on a different actor and this motion could be used as a driving signal. Run and side shuffle have smaller errors than the curved walk, but for detailed interaction, these motions would probably also not provide sufficiently accurate z-depth values.

This evaluation shows that a motion capture segment that works well provides a reasonable prior for both root motion and relative depths of different limbs. In the case of the curved walk, the large error in z-depth occurs because the motion of the root was different from ground truth. In the case of the side shuffle, root motion was quite similar to the ground truth (straight line, maintaining almost the same depth from the camera for the entire duration). However, the relative depths of the limbs were completely different. The happy walk provided a reasonable approximation to both of these quantities.

In Section 2.3.4, we reformulate the input-match term e_a to align the 3D proxy to the modified motion capture pose, rather than the original hand-drawn pose. The modification is intended to transfer the style of hand-animation onto the 3D proxy, while filtering out noise. Here, we evaluate the question: was the modification successful, or are we simply playing back the motion prior? Numerically, the modification should cause the 3D proxy to be a better match to the artist-drawn animation than the motion prior.

The input is a motion capture walk (‘normal walk’) projected to 2D. The motion prior is chosen to be a stylistically different walk (happy walk) captured on a different subject. We compare the distance between markers on the ground truth with markers on the the motion prior and the 3D proxy. The root mean squared (RMS) error for each virtual marker is shown in Figure 2.22. The ticks on the x-axis represent the virtual markers. Ticks 1 through 42 are markers on the ‘upper body’, and Ticks 43 through 72 represent markers on the ‘lower body’. The RMS error between the ground truth marker positions and the 3D marker positions of the ‘happy walk’ motion prior, averaged over a 100 frame segment, are shown in red. Because the limb segments belonging to the upper body are modified to match the pose descriptor of the synthetic input markers, we can see that the error for these markers is reduced in the 3D proxy (blue), while the error for the lower body markers is unchanged, as expected.

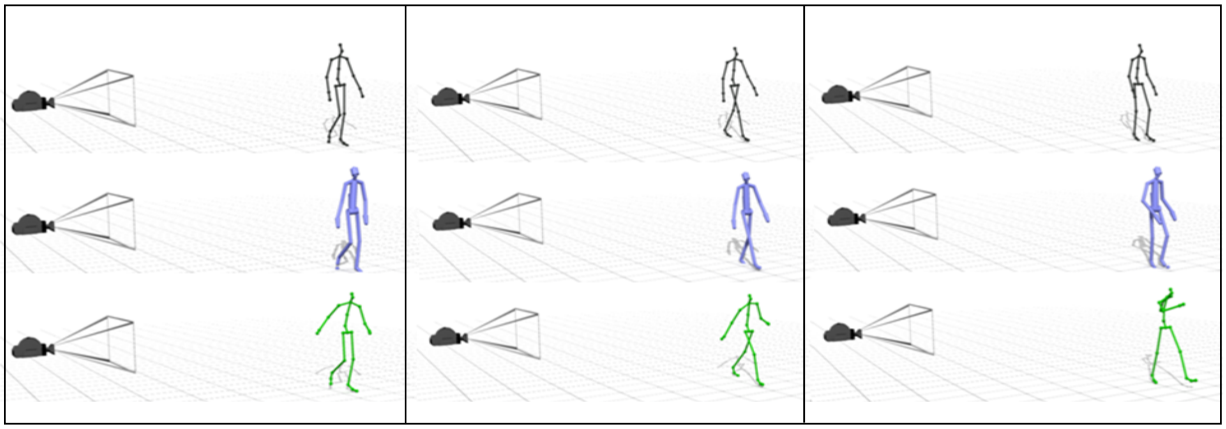


Figure 2.20: We compare the ground-truth walk (black, top row) with our result (purple, middle row), and the motion capture segment which provided z-depth information (green, bottom row). This motion capture segment is a 'happy walk', where the actor walked in the same direction but swung their arms and legs higher to communicate happiness. Even though the 3D marker positions for the motion capture poses are quite different, the result is close to the ground-truth pose because the motion capture poses provide only depth.

2.6 Discussion

In this work we have shown that connecting hand-drawn animation with 3D computer animation offers many advantages—simulating physically driven clothes for a hand-drawn character, for example, or empowering traditionally trained animators to contribute to a 3D character's performance. This connection is made by generating a 3D representation, or proxy, for the 2D hand-drawn character, and allowing the proxy to interact with the other scene elements in the 3D virtual world. We observe that the viewer cares about different quantities depending on the function of the 3D proxy of the hand-drawn character. For example, when the 3D proxy serves as an attachment point, an image-plane mismatch between the proxy and the hand drawing will draw the viewer's attention because the 3D simulation will no longer appear to be attached to (or driven by) the hand-drawn character. In contrast, for a joint hierarchy that is skinned and rendered from different camera angles, an image-plane mismatch attracts less viewer attention than a limb intersecting with another limb. Therefore, we create different levels of detail in the 3D proxy for different levels of interaction between the hand-drawn character and the 3D scene elements.

Because we need a contiguous motion capture sequence to infer z-depth, this approach is limited to hand animations of human-like characters, and the amount of flexibility in computing motion clips provided by state-of-the-art time warping, retargeting, motion blending, and motion re-sequencing techniques (using, for example, [7, 39, 49, 64, 131]). Three-dimensional keyframe sequences for non-humanoid characters would work equally well if such information was available for a character. A useful direction of future research would be to incorporate motion models as a source of depth information rather than a contiguous sequence of motion capture poses.

In Section 2.3.4, we minimized the linear system in a low-dimensional subspace and then

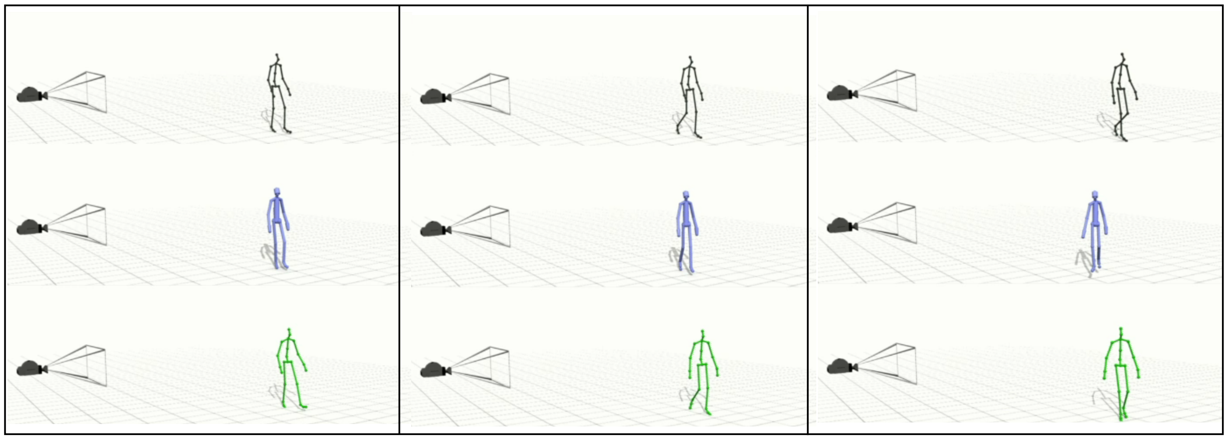


Figure 2.21: We compare the ground-truth walk (black, top row) against our result (purple, middle row), when z-depth information is provided by a 'curved walk' motion capture segment (green, bottom row). Because the motion capture poses curve towards the camera, the z-depth values are not the same as the z-depth values for the ground-truth poses. We can see that our result also curves towards the camera because the result poses have the same z-depths as the motion capture poses.

projected it to full space to obtain the 3D world positions for the virtual markers. We investigated how the solution would change when the number of principal components P was increased. More dimensions allow us to capture finer details, as can be seen in the elbow and wrist angle in Figure 2.23. This motion prior keeps the solution well-behaved even in the full-dimensional space.

Our method does not include non-penetration constraints as part of the error terms in the minimization. As a result, it is possible for the limbs of the 3D proxy to interpenetrate each other. Using well-behaved motion capture data as a motion prior alleviates this problem, but does not guarantee non-penetration. Simulating tight-fitting clothes, such as a shirt with sleeves, might be difficult as a result of this limitation.

The user specifies the camera moves implicitly, by selecting a motion capture segment. Thus, if the database contains only an overground walk cycle, and the hand-animation contains the character walking in place (on a treadmill), our method will assume that the character walks overground with a tracking camera. The water balloon result in Figure 2.16 is an example that stress-tests the assumption that the best-matching motion capture segment will be sufficient to estimate the camera imagined by the artist. In the animated movie (submitted as supplementary material), we see that the estimated camera tracks the character as he walks, but moves very slightly instead of remaining completely stationary when the character ducks. This movement happens because it is not reasonable to find, or capture, a motion capture segment where the actor has ducked with the arms and legs bending exactly in the same order as in the hand animation—for example, the actor might brace their arms over their head before they bend at the knees. As a result, our camera estimation routine incorrectly infers that the camera should move to account for this mismatch. Future work could incorporate information about camera moves from the shot exposure sheet or add annotation about the world coordinate frame in the process of animating. A possible annotation could be asking the artist to draw a cube on the ground plane for every

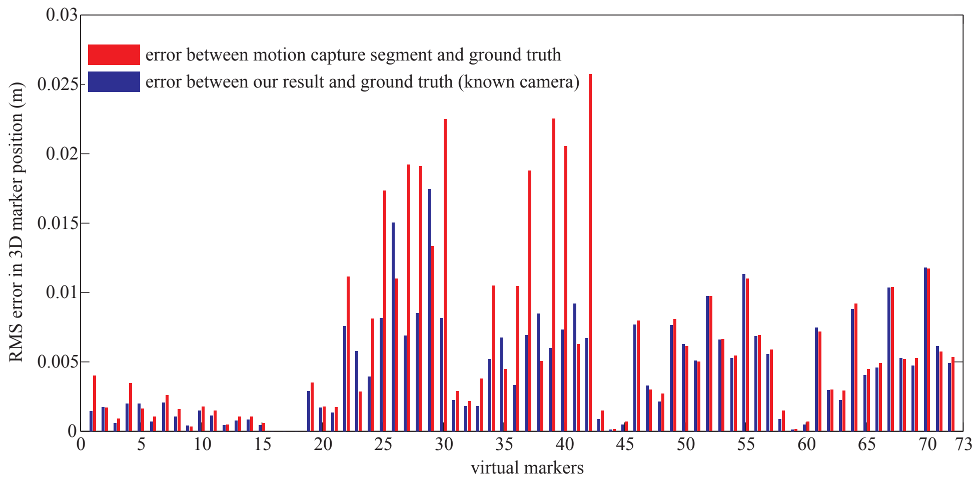


Figure 2.22: Error in 3D world positions for virtual markers when motion capture data is modified via the pose descriptor.

frame, thus providing a world reference.

Figure 2.1 illustrates that the level of detail in a 3D proxy depends on its function. It would be interesting to extend this approach to generate a mesh model that conforms to the artist's lines, thus allowing 3D texture-mapping and lighting techniques to be used for hand-drawn animation. Li and colleagues have discussed a method to deform a mesh model once the driving skeleton has been modified to match the hand-drawn animation [65]. We could use the method of Section 2.3.4 to match the driving skeleton to markers on the hand-drawn animation, thus automating one of the user inputs in that method. Their method handles deformation fields only in the image plane. Future work in shape understanding could allow the 3D proxy to squash and stretch out of the image plane as well.

The pose descriptor in Section 2.3.2 can be thought of as a shape descriptor that retargets the pose of the hand-drawn figure onto a skeleton proportioned to match the motion capture data. This step allows us to transfer the pose from a hand-drawn ballerina, who may be statuesque, onto a 'normal' female actor, or the pose of a goofy character with long arms onto a 'normal' male actor, and then use human motion capture data as a motion prior. Our pose descriptor preserves in-plane joint angles in its current formulation, and it would be interesting to extend it to preserve contacts or other metrics that might better capture the essence of the hand-drawn animation.

Though all our results have used hand-animations as input, we could apply the same concepts to 3D proxies for video sequences—for example, to add a scarf on a person walking outdoors on a windy day. Just as we have matched the rendering style of the added 3D elements by using a toon shader, we could draw on the literature in the vision and computer graphics communities on transferring lighting, shadows and other visual cues, so as to augment video data with 3D computer generated elements.

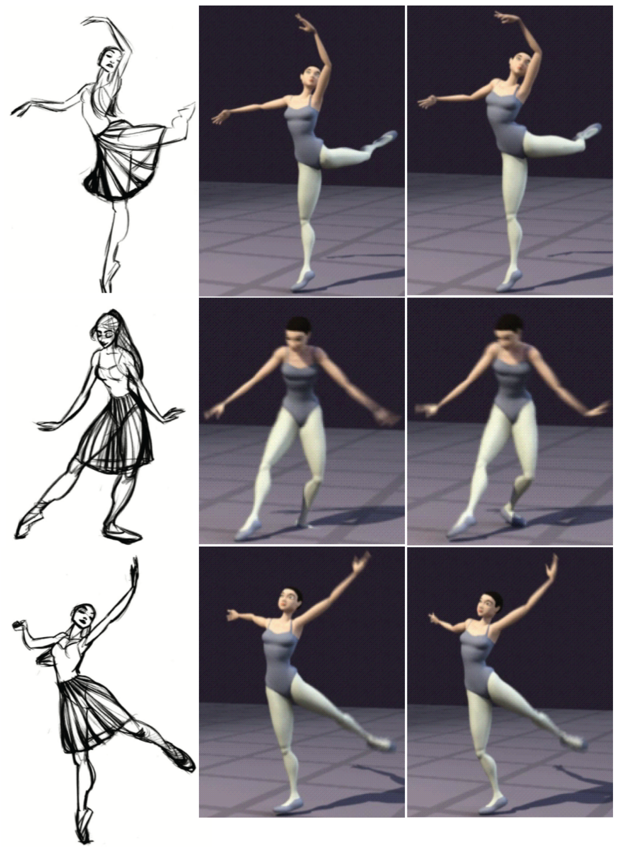


Figure 2.23: Projection to a low-dimensional subspace: We varied the number of principal components P used to represent the 3D pose. Left: hand-drawn frames. Center: $P = 20$. Right: $P = 72$ (all dimensions). More dimensions allow for finer details, as seen in the elbow and wrist.

Chapter 3

Augmenting Stills with Camera Moves based on Gaze Data

3.1 Introduction

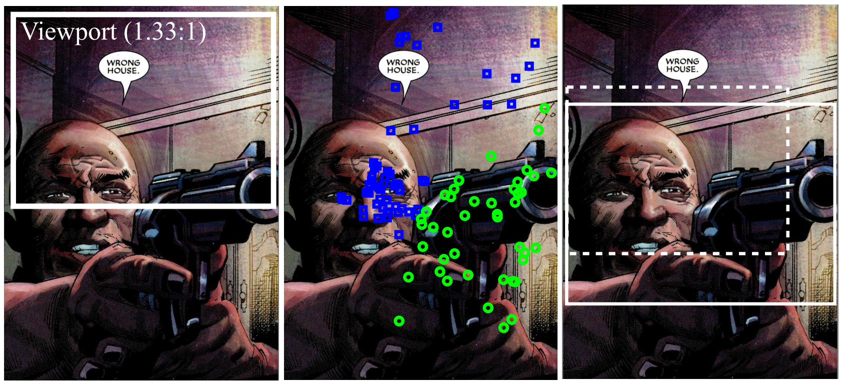
Integrating stills into motion pictures via camera moves is often known as the *Ken Burns* or *Ken Morse effect*, in different parts of the world. This effect was traditionally used to engage the viewer when photographs or paintings were shown on television or film. Today, a camera move on a still picture has an additional role to play: it allows original format content to be retargeted to different digital display devices. Presenting a still picture as a sequence of images seen through a viewport allows us to view pictures with varied aspect ratios on a fixed size screen (Figure 3.1 (left)). For comic book art in particular, moves-on-stills offer the advantage of engaging a strength of digital displays (the presented material can be animated) while maintaining the defining characteristic of sequential art (each panel is a moment frozen in time).

Moves-on-stills used to be created by a camera operator filming the still with a rostrum camera on an animation table. Today, software such as Adobe Premiere and iMovie allow anyone with a computer to create this effect digitally. While the mechanics have been made simpler, a central question remains unexplored: what makes for a good camera move on a still? For example, if the picture contains a man looking at an object he holds in his hand, the camera move should probably present the man's face, and the object in his hand, either moving downward from the face, or upwards from the object. Will Eisner points out that for comics, "...the artist must...secure control of the reader's attention and dictate the sequence in which the reader will follow the narrative..." [32]. Therefore, a good camera move should respect this artistic intent.

However, generating a camera move, especially one that supports the narrative in the original still picture, is a complex process: it involves guessing at the artist's intent by recognizing objects such as 'man' and 'hand', understanding relationships such as 'looking at' and figuring out how these relationships influence a camera move. We do not yet have computational models to replicate this complex cognition. Randomly picking a start position and an end position for the camera would result in a large number of possible camera moves, most of which are meaningless, some that actually contain the objects of interest, and only a few that support the narrative. Our key insight is that comic book artists make a conscious effort to direct the viewer's gaze, therefore, recording the gaze data of viewers should tell us the visual route that was used to discover (and conversely, may be used to present) the narrative in the picture. Figure 3.1 (middle) shows the viewer gaze data collected on an example comic book panel.

The first challenge posed by this insight is that eyetracking data is a noisy signal. The eye movements of viewers are governed both by characteristics of the visual stimuli and by individual-specific viewing strategies. As a result, no two viewers will have identical eye movements on a given visual stimulus. This inter-observer variability can be thought of as process noise. Additionally, imperfections in the eyetracking device such as calibration error and dropped data lead to measurement noise.

The second challenge is that eye movements do not directly map to camera moves. The eye movements used to traverse still images are saccades,i.e., rapid movements of the eye to center the fovea at a region of interest; fixations are the eye resting in the region of interest between saccades [81]. Repeated sequences of saccades, called 'scanpaths', are involved when an observer visual processes a given stimulus image [79]. While eye movements occur to efficiently process visual information, camera moves are intended to present information. In the example of the man holding the object, if the camera were to move as rapidly as the eye does, from the face of



Comic book panel (1:1.37) Viewer gaze data (two clusters) Our result for move-on-still based on gaze data

Figure 3.1: A move-on-still moves a viewport (e.g., 1.33:1) over original artwork (e.g., 1:1.37) to best display the art. Our key insight is that viewer gaze can be used to predict the parameters of a move-on-still, if comic book artists are successful in purposefully directing viewer attention. We record gaze, verify eye movement consistency across observers, and demonstrate an algorithm to compute move-on-still-parameters from gaze data.

the man to the object in his hand and then back and forth, the resulting video would be jarring to look at and difficult to follow.

Our first contribution is to show that process noise is reduced as a result of purposeful direction by comic book artists, that is, the artist is able to direct the viewers' attention through the picture. We collect gaze data from human participants and find greater consistency in eye movements for comic book art versus photographs that were not taken with the expressed intent of directing viewer attention. Our second contribution is an algorithm to predict the parameters of a virtual camera move from viewer gaze data. We convert the raw gaze data into fixation locations, saccade directions, and dwell times. Features extracted from this data are used to probabilistically predict the parameters of a virtual rostrum camera move (Figure 3.1 (right)).

3.2 Background

A challenge in processing eye movements is that they are driven by both the properties of the presented stimuli (bottom-up cues) and the idiosyncrasies of the viewer (top-down influences). However, artistic wisdom is that photographers, graphic artists and film directors are able to get viewers to look at certain regions versus others. Researchers have recognized this difference in the visual perception of artistic stimuli versus 'natural' stimuli and investigated this effect. Dorr and colleagues found that the characteristics of eye movements on static images are different from those on continuous videos, and the movement patterns in Hollywood movies are significantly more coherent than natural movies (the beach scene in their paper, for example) [68]. Carmi looked at the effect of 'MTV-style jump cuts' compared with continuous video in reducing inter-observer variability in eye movements [13]. Omori and colleagues recorded eye movements on Japanese manga to understand what why readers skipped certain panels and paid

(a) Amateur snapshots (b) Robot pictures (c) Remote infra-red eyetracker

Figure 3.2: Sample photographs from the control set composed of (a) amateur snapshots and (b) robot pictures. We found that the eye movements across viewers were more similar for comic book pictures compared to this control set. (c) Gaze data was collected with a remote infra-red eyetracker.

attention to others [116]. We contribute to the research in the visual perception of artistic content by experimentally verifying a widely held belief in the sequential art community: that the artist is able to purposefully direct the visual attention of readers through the pictorial narrative (see Figure 3.2).

Computer graphics researchers have recognized that gaze is a source of high-level information about the underlying scene. Howlett and colleagues investigated if three-dimensional models can be simplified based on where people look [47]. DeCarlo and colleagues observed that people fixate on regions of semantic importance, therefore, gaze data can guide non-photorealistic rendering [24] and image cropping [106]. Recently, Jacobs and colleagues proposed the notion of 'cosaliency' to refer to the salient components of an image for the purpose of comparison with another image [31]. They train a model of 'cosaliency' based on human gaze data and use it to predict crops for collections of images. Not only is 'looking at' a photograph easier and more fun than mouse clicking, it allows us to access what is important to the viewer without interference by the participant's own introspection. We build upon this insight by observing that, in addition to the attended locations, the order in which people attended to the locations and the duration they spent looking is also informative of the underlying pictorial composition.

Sequential art is made up of visual elements arranged to convey information and/or produce an aesthetic response in the viewer [70]. Each panel in a comic book is a moment frozen in time. Shamir and colleagues provide an algorithm to choose the appropriate time points to create comic panels that summarize an interactive game play in the graphic novel style. There are also several works that automatically extract individual panels from the page layout [5, 15, 45] and change panel size based on the corresponding narrative time [8]. Our work is novel in this space as we consider the problem of retargeting comic panels to digital displays via moves-on-stills.

Past work in retargeting photographs includes the method by Chen and colleagues to adapt photographs to a small display by centering the viewport in the area of maximum interest, as identified by features such as contrast, edges and face detections [16]. This method relocates the viewport from the center of the image to the interesting region of the image, but does not create a trajectory for the viewport, as needed for moves-on-stills. Zheng and colleagues add 3D pan-and-scan effects to still photographs by capturing multiple viewpoints, reconstructing depth and using depth information to select the camera move (a move that best shows off parallax) [132]. This method does not apply to single images such as comic panels because of the need for depth reconstruction.

There has also been research in improving the accessibility of camera control interfaces [30, 40, 53], and towards automatic algorithms to plan the movement of the camera [29, 43, 69]. This work operates inside a computer-generated virtual environment, where all the elements in the scene are known to the system, for example, character A is talking to character B. These methods do not directly apply to still pictures because RGB images do not come annotated with the required high-level information such as who is talking and who is listening. In our work, we show that instead of a mouse based interface to move a camera or annotate the image, recording the eye movements of viewers provides all the information needed.

While our work uses gaze data to drive a computer-generated camera move, human-computer interaction research has explored whether points of regard are an indicator of higher-level mental states such as 'interest', 'understanding', and 'confusion'. Starker and Bolt envisioned a storyteller that changes the subject of its conversation based on where the viewer is looking and zooms in towards an object when talking about it [114]. Recently, gaze data was used to estimate which objects are of interest to a viewer for augmented displays [57]. Conati and colleagues infer how well a student has understood a tutorial based on their gaze pattern while interacting with an interactive learning environment [18]. Hyrskykari and colleagues present an eye-aware translation aid which provides help when the user's gaze path indicates that there is difficulty in reading [48].

3.3 Eyetracking Viewers on Various Image Categories

Our hypothesis, that we can access the artist-designed visual route through an image by recording viewer gaze, is based on the assumption that the artist was successful in directing viewers' attention through a picture. We verify this assumption by posing the following two-part question: if there was no effort put into directing the viewer's eye through the picture, then how much eye movement consistency do we observe across different viewers? If there is expressed effort put into directing the viewer's attention, then, is the consistency across viewers increased?

We showed viewers a collection of pictures drawn from five categories: 32 images from the *Watchmen* graphic novel, 35 images from the *Ironman: Extremis* comic book, 18 photographs from photoessays found in online issues of political and nature magazines, 30 images from the collection of 'robot' pictures and 30 images from the collection of 'amateur snapshots' (sample images are shown in Figure 3.2 (a) and (b)). 'Robot' pictures were collected by a robot wandering through the hallways of a campus building during an automatic data collection project [58]. The amateur snapshots were randomly chosen from the online dataset made available by Judd and colleagues [56]. Judd described the images in the database as 'amateur snapshots' (personal communication), which refers to the quality of the photographs as being amateur records of events, rather than carefully composed works of art. In total, every participant observed 145 pictures.

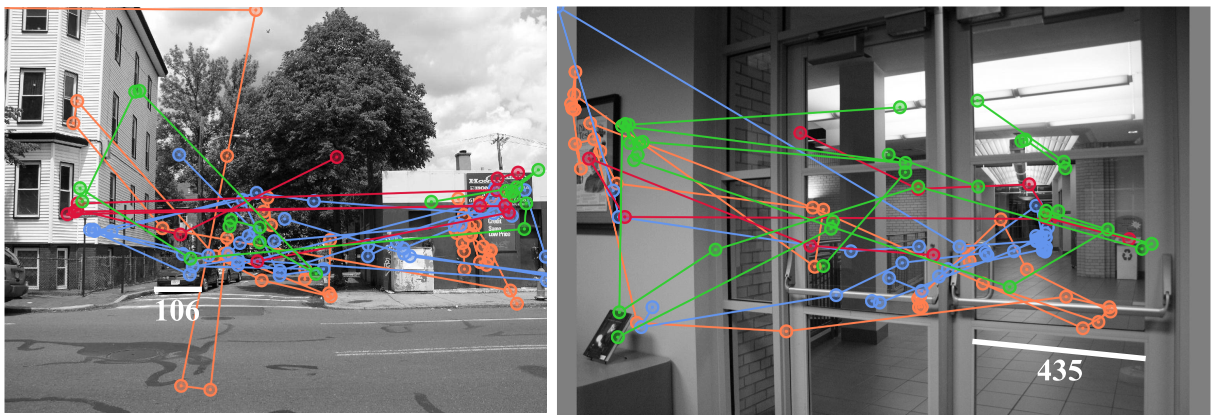


Figure 3.3: Eyetracking data from four participants is overlaid on a stimulus image. The circles represent fixations and the lines represent saccades. The stimulus image is made as large as possible on a 1680×1050 screen. Left: a sample image from the 'amateur snapshots' category. The width of the car is roughly 106 pixels. Right: a sample image from the 'robot' category. The width of the door is approximately 435 pixels.

Method

Five naïve participants (4 males, 1 female, age ranging from 21 to 44 years) were recruited from a participant pool available via a website, in accordance with the guidelines set by the institute's review board. They were compensated monetarily for their time. The data for one participant was removed because of large calibration error in the eyetracking device.

Participants were asked to view the stimulus images and told that they would be asked ten questions at random points in the viewing session. The questions were based on comprehension, rather than memorization, for example, 'What did the man find in the cupboard?' and 'Was there a fire extinguisher in the building?', and the answers were multiple choice. The stimuli images were presented in randomized order. We chose not to advance the stimuli automatically because it would be impossible to set the timing appropriately for every picture and every participant. Too little time might result in the picture advancing too soon, for example, the participant might be reading the text in the word bubbles and would then miss seeing the pictorial portion. On the other hand, it has been observed that consistency in fixation locations decreases as more time is spent on the stimulus, probably because top-down strategies diverge [118], and therefore, too much time could result in individual idiosyncrasies being asserted over artistic direction. To balance these constraints, we paced the study so that participants could advance through the images by pressing a button, but were required to spend a certain minimal amount of time on each stimuli image. The minimal time was set to be the average time spent by the pretest participants (four seconds). With this minimal time setting in the experiment, we found that participants spent on average 9.62 seconds per stimulus image ($\sigma = 3.74$).

Participants viewed the stimuli on a 22 inch monitor, at pixel resolution 1680×1050. All images were resized to be as large as possible, while keeping their aspect ratio intact. They were letter-boxed with a gray matte (RGB$= (127, 127, 127)$) to fit the stimuli presentation screen. The

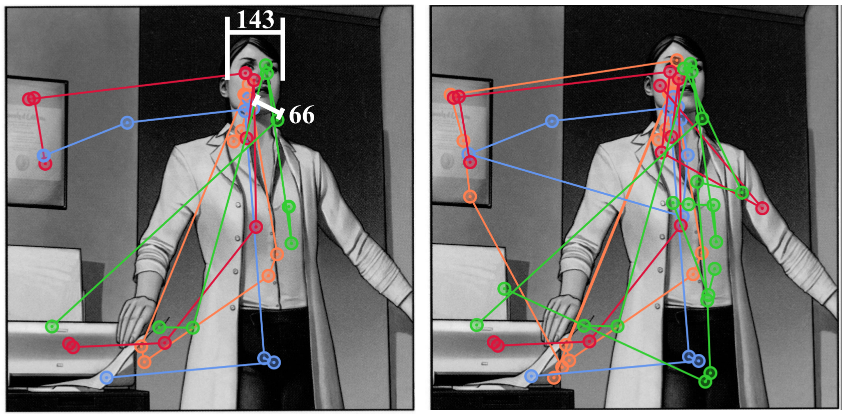


Figure 3.4: Eyetracking data from four participants is overlaid on a stimulus image. The circles represent fixations and the lines represent saccades. The stimulus image is resized to fit to a 1680×1050 screen. The width of the woman's face is about 143 pixels. Calibration error is $30 - 40$ pixels. Left: eyetracking data for the first three seconds. Right: eyetracking data for the full duration.

participants sat approximately 18-24 inches away from the screen. A visual angle of 1 degree was approximately 30 pixels at these settings. Figures 3.3 and 3.4 illustrate the dimensions of various parts of the stimuli images as a comparison.

Participants were asked to adjust the chair to a height and distance comfortable to them; then, the system was calibrated. After calibration, they could move their head freely. This setup allowed for a natural setting in which to observe the presented materials, unlike systems that require a chin rest or a head-mounted eyetracker. The participants all had normal vision or wore contact lenses to correct to normal vision. For eyetracking, we used SensoMotoric Instruments' RED eyetracker, running the iViewX software (Figure 3.2(c)). Points of regard, that is, the two-dimensional gaze locations on the calibrated screen, were recorded at 60 Hz. Raw data was converted into saccades (rapid eye movements between objects of interest) and fixation points (the eye is positioned so that the object of interest is in the foveal region) inside the proprietary SMI analysis software before being exported as a spreadsheet.

Analysis

Our goal is to discover numerically if the viewers who were eyetracked followed a more similar visual route to each other while viewing stimuli belonging to the comic art categories, compared with the robot or amateur pictures. We analyzed similarity in the unordered and ordered fixation locations of observers across categories. Because we are interested in understanding the spread of visual attention through the scene, we compute fixation locations and fixation order, but not fixations durations, in our analysis.

Several metrics have been proposed to compute the similarity between eye movements of different observers. One class of metrics constructs a fixation map from recorded gaze data by convolving the fixation locations with Gaussian kernels [85, 128]. The fixation map for one observer is thresholded to create a binary map, which serves as a classifier for the fixation locations of all other observers [56, 118]. The percentage of fixations that fall inside the map are the inliers and the percentage of total area inside the binary map is the salient percentage for a given threshold value. An ROC curve is plotted by varying the value of the threshold. The mean curves for each stimuli category, by leaving out each observer in turn, are shown in Figure 3.5 (a).

The area under the ROC curve measures how similar the fixation locations of a given observer are to other observers. Curves closer to the diagonal indicate lesser similarity compared with curves that rise sharply. Because comic books typically contain text inside word bubbles, while photographs do not, we plot ROC curves for the comic book categories with text (solid lines with cross markers) and without text (by color segmenting out the word bubbles). The curves for *Watchmen* are nearly identical with and without text because only four of the 32 images contain word bubbles, with only one word each.

The analysis of the ROC areas (Figure 3.5 (b)) shows that fixation locations are more consistent across observers for the comic book categories, *Ironman* and *Watchmen* (red and magenta curves, respectively), compared to the robot pictures, amateur snapshots and photoessays (green, dark blue and cyan curves). The difference in ROC areas is significant (one-way ANOVA, $F(4, 140) = 31.35, p < 0.05$, followed by Tukey-Kramer for post-hoc analysis, $p < 0.05$).

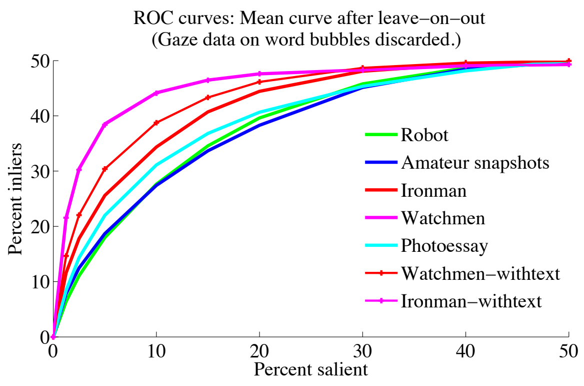
We also verified this analysis with another measure: Normalized Scanpath Saliency (NSS) [68, 97]. The fixation map is normalized to zero mean and unit standard deviation. For each observer, the NSS score is computed by looking up the values of the fixation map (for all other observers) at the fixation locations of this observer and taking the average. If this observer had fixated on locations similar to other observers, the average value will be larger than if this observer had fixated on different locations. In Figure 3.5 (c), we show the NSS score for each category. For our analysis, we used a Gaussian kernel with $\sigma = 50$, which is about $1°$-$1.5°$ visual angle.

We computed the NSS score with each participant in turn as the test observer (leave-one-out) and averaged this score to obtain the NSS measure of inter-observer gaze variability for each stimulus image. Gaze data from different observers was significantly more consistent on *Watchmen* and *Ironman*, compared with robot pictures, amateur snapshots and photoessays (one-way ANOVA, $F(4, 140) = 9.26, p < 0.05$), followed by Tukey-Kramer for post-hoc analysis, $p < 0.05$).

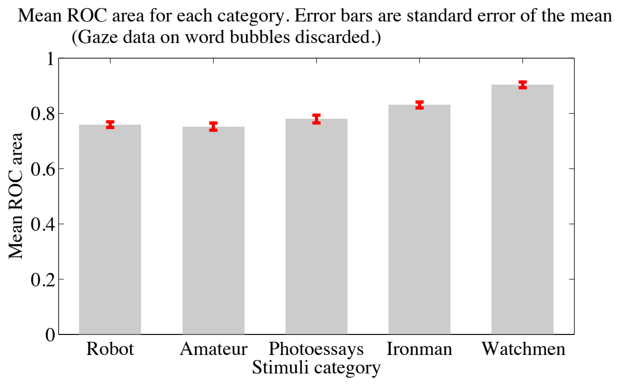
These two metrics measure the spatial similarity in fixation locations, irrespective of the order in which the fixations occurred. If an observer looked at the top left corner and then the bottom right corner, and another observer looked at the same two locations but in the opposite order, they would still have a high similarity score. Because artists aim to deliberately direct the viewer's attention through their picture, we expect that observers will not only look at similar locations, but also do so in the same order.

The second class of metrics to measure similarity in eye movements considers both spatial and temporal similarity, that is, fixation locations and their order. As an example, the stimulus images may be preprocessed to mark out regions of interest (ROI's) which are assigned unique alphabet identifiers. Eye movements for different observers are compared by computing the string-edit (Levenshtein) distance between strings obtained by concatenating ROI identifiers. Because this metric requires us to know the regions of interest a priori, it is suitable for a study where there is a well-known structure to the stimuli (e.g., the presence of a face).

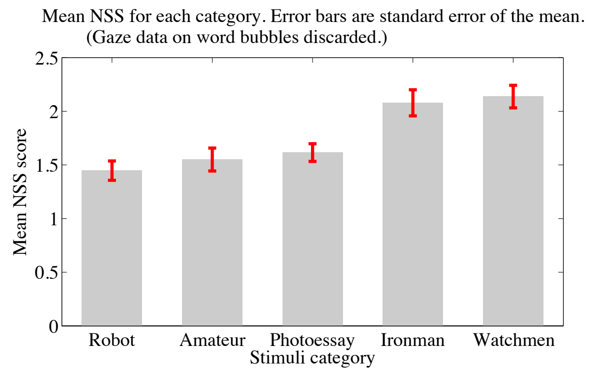
Instead, we follow the method by Jarodska and colleagues, who expressed recorded gaze data as vectors of positions and durations and presented a suite of vector distance metrics to



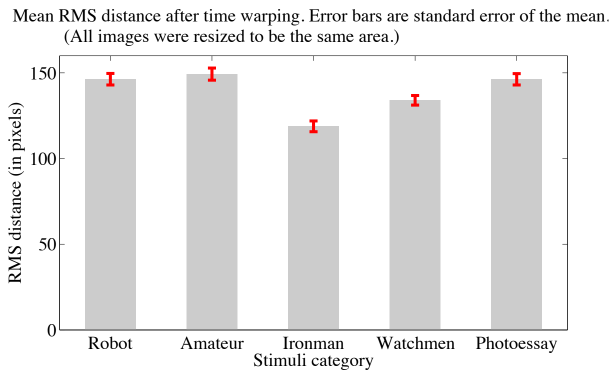
(a)



(b)



(c)



(d)

Figure 3.5: Viewer gaze data is more consistent for the comic book panels compared with the robot pictures, amateur snapshots and photoessays. (a) ROC curves for the different categories. (b) Mean ROC area for each category. (c) Mean Normalized Scanpath Saliency (NSS) score for each category. (d) RMS distance between ordered fixation locations.

compute inter-observer consistency [52]. We denote the ordered fixation locations for observer s on picture p by $\mathbf{X}^{ps}$. Then, $\mathbf{X}^{ps} = [\mathbf{X}_1^{ps}, \mathbf{X}_2^{ps}, \dots, \mathbf{X}_{\gamma_{ps}}^{ps}]^T$, where $\mathbf{X}_l^{ps}$ is the l-th fixation location, and γ_{ps} is the total number of fixations on this picture. The value γ_{ps} could be different because two observers might have different reading speeds, or, because one of them fixated longer at each location, while the other jumped back and forth between the locations, for example. In addition, the same observer might attend to a different number of regions on different stimuli, or, change their speed of perusal.

We warp the vectors $\mathbf{X}^{ps}$ to be the same length via the Dynamic Time Warp algorithm [34, 105]. Let $\tilde{\mathbf{X}}^{ps}$ be the warped gaze path for observer s on the picture p, and $\tilde{\gamma}_{ps}$ be the length of this vector. The distance $\Delta_{i,j}^{p}$ between the gaze paths of observers i and j on the picture p is the root mean squared Euclidean distance in image space between the warped fixation location vectors.

$$\Delta_{ij}^{p} = \sqrt{\frac{||\tilde{\mathbf{X}}^{pi} - \tilde{\mathbf{X}}^{pj}||_2}{\tilde{\gamma}_{pi}}} \quad i, j = 1, 2, \dots, S, \tag{3.1}$$

where S is the total number of observers in the experiment. This measure captures spatiotemporal variability and is similar to the vector comparison metric presented by Jarodska and colleagues when the vector consists of fixation locations only [52].

Figure 3.5 (d) illustrates the mean distances for each of the five categories. The mean distances for the comic books *Ironman* and *Watchmen* are statistically different from the robot dataset and the amateur snapshots (Tukey Kramer test, $p < 0.05$). We see greater inter-observer consistency for *Ironman* compared to *Watchmen* (Tukey Kramer test, $p < 0.05$), most likely because the pictures in the *Ironman* set have word bubbles with a fair amount of text. Only four pictures in the *Watchmen* set have word bubbles, and they contain exactly one word each.

In all the analyses, eye movements across observers are more dissimilar for the photoessays compared with the graphic artwork. This difference could be because photographs, however carefully composed, are still tied to the physical world. Photographers have limited control over many aspects of the scene they are capturing, for example, it may be difficult to avoid a distracting billboard in the background. The painterly style of the graphic art, involving strong contrast, sharp outlines and bold colors, is also different from the photographs. It is likely that graphic artists manipulate the painterly effects to direct the viewer's attention, something that photographers have limited control over. Finally, photographers are limited to a physical camera, so they cannot exaggerate too much, for example, by enlarging a person's fist more than their lens will allow.

3.4 Moves-on-stills

Creating moves-on-stills involves cognitive processes such as object recognition and scene understanding. We do not yet have good computational understanding of these sophisticated processes. Our key insight is that comic book artists make a deliberate effort to direct viewer attention through their images; if they are successful (as we have experimentally verified), then the viewer gaze data on given images can be used to predict the parameters of moves-on-stills on those images.

Figure 3.6 shows the raw gaze data in the top row. The magnitude of eye movement in the x-direction is less than the movement in the y-direction because the panel is tall and narrow. Because different viewers take different durations of time to read a panel, we time warp each person's raw data to match the person who spent the longest duration (Figure 3.6 middle row). It is apparent that the viewers follow similar (though not exactly same) visual routes through this panel. We also see the repeated scanpaths, which are a characteristic of eye movements on images first observed by Noton and Stark [79]. Though the human perceptual system uses fast and recurring eye movements to process the image, if a camera were to follow a similar trajectory, the resulting movie would be disturbing to the viewer.

In Figure 3.6 (bottom row), raw gaze data is converted into fixations and saccades inside the proprietary eyetracking software. Saccades are the rapid, ballistic eye movements that cause the fovea to be focused at a new location, and fixations are the events between saccades during which the eye stays foveated at a location. Though the structure in the data is not discernable in the plots of the x and y coordinates versus fixation number, it is apparent in the x-y plot overlaid on the comic panel. Our algorithm computes the parameters of a move-on-still from the clusters of fixation locations, associated saccade directions and fixation durations, and uses the

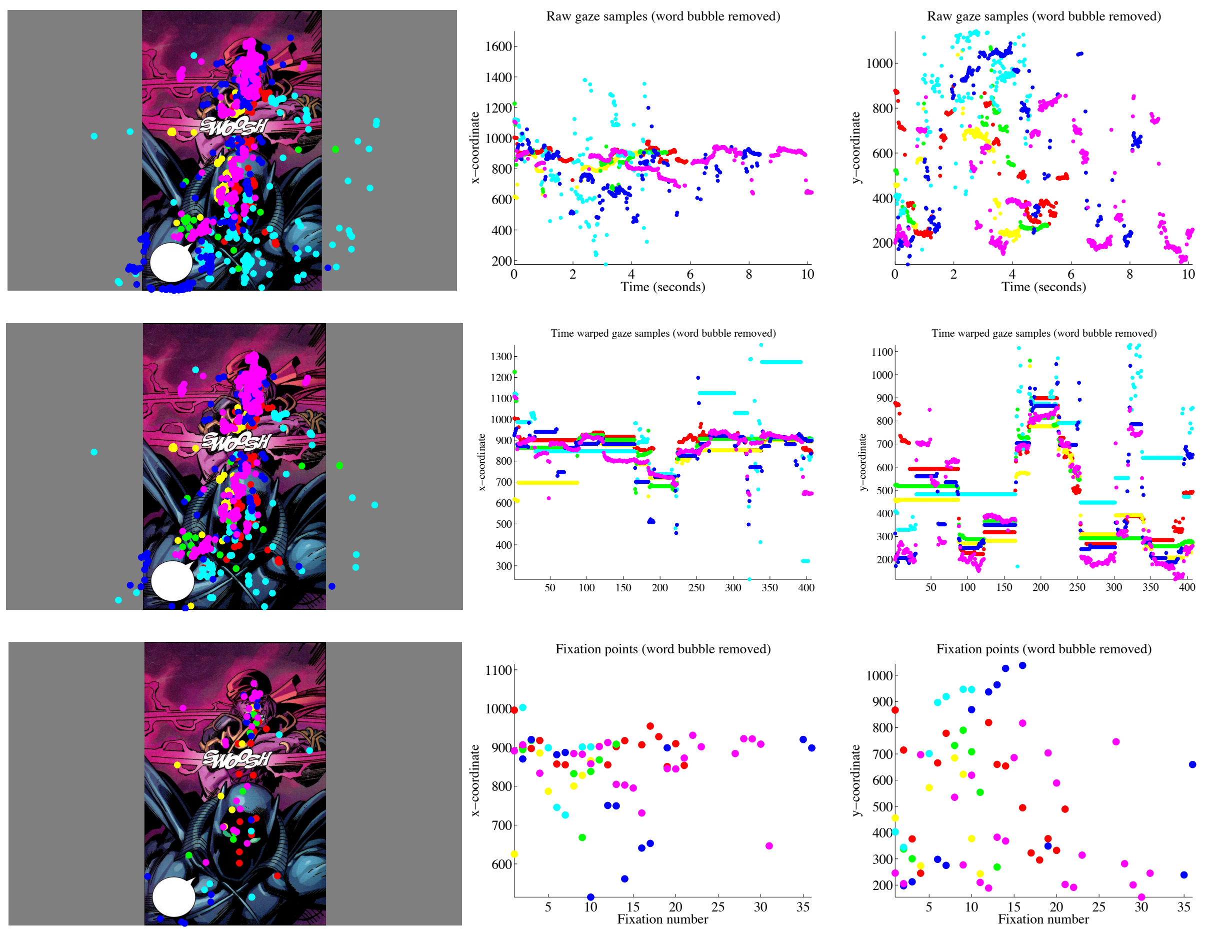


Figure 3.6: Various views of the data recorded on a comic panel.

compositional guideline known as "rule of thirds" to resolve a final framing ambiguity.

Moves-on-stills were traditionally filmed on a rostrum camera and animation table. The degrees of freedom for a digital move on a still are the same degrees that were afforded by the physical setup. There is one degree of vertical movement, towards or away from the animation table, that yields a *track in* or *track out* shot. There are two degrees of movement on the animation table, North-South and East-West, that result in a *pan* shot. The fourth degree of freedom, rotation of the artwork about the vertical axis, is rarely used for comics, and therefore, we focus on the translations only. In the digital setup, these degrees of freedom are characterized by the position of the center of the viewport (or framing window), and the dimensions of the viewport around the point of interest.

3.4.1 Points of interest

Points of interest are the semantically meaningful elements of the picture, which the artist used to tell the story, and which the rostrum camera operator would like to frame through the viewport. In the example image in Figure 3.4, the points of interest are the face and the hand. Because

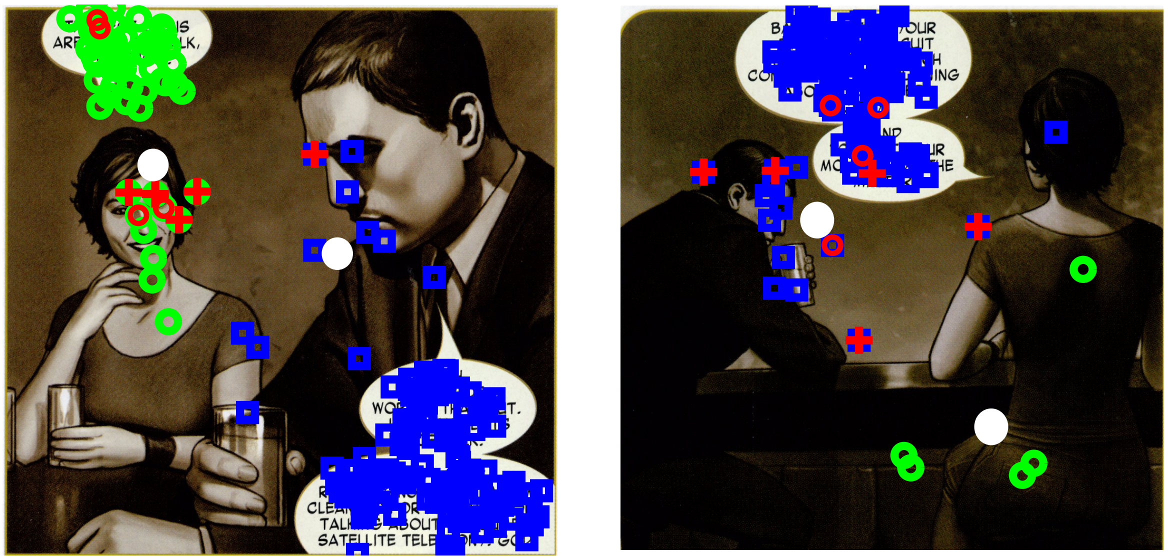


Figure 3.7: Left: noise in data capture leads to reading fixations that are outside the word bubble. Right: reading and non-reading fixations can be quite close to each other, as in the case of the word bubble and the man's head.

visual information processing occurs during fixations [81], the locations of the recorded fixation points indicate regions of important semantic content. We discover the points of interest in a still picture from the spatial distribution of fixation locations.

Unsupervised clustering is performed on the locations of the fixation points, based on squared Euclidean distance. Let each fixation location $\mathbf{X}_l^{ps} \in \mathcal{R}^2$ be associated with its cluster via the label $z_l^{ps} = k, k = \{1, 2, \ldots, K\}$ where K is the total number of clusters and Ω_k is the set of all fixations belonging to the cluster k. Then, the point of interest μ_k is the centroid of all the fixation locations belonging to a cluster Ω_k,

$$\mu_k = \frac{1}{|\Omega_k|} \sum_{s=1}^{S} \sum_{l=1}^{\gamma_{ps}} \mathbf{X}_l^{ps} \delta(z_l^{ps} - k), \tag{3.2}$$

where δ refers to the Kronecker delta function. The dwell time τ_k on a point of interest is the total duration of all fixations in its cluster,

$$\tau_k = \sum_{s=1}^{S} \sum_{l=1}^{\gamma_{ps}} t_l^{ps} \delta(z_l^{ps} - k), \tag{3.3}$$

where t_l^{ps} is the duration of each fixation.

Fixations on word bubbles are not usually informative of the points of interest because word bubbles are placed to avoid occluding a pictorial element. Accordingly, we color segment the word bubbles, identify the fixations which lie inside the segmented region, and remove them.

In practice, eyetracking data is noisy and it is not straightforward to identify the word bubble fixations. These fixations may lie outside the colored word bubbles because of calibration error.

For example, the green colored fixation points in Figure 3.7 (left), lie slightly outside the word bubble, and are a source of error in the computed centroid (white circle). Sometimes though, the fixation points right outside the word bubbles might not be 'reading fixations', as in Figure 3.7 (right), where the fixations on the man's face are about as far away from the word bubble as the erroneous data. Thus, simple heuristics like dilating the color segmented word bubble or identifying reading fixations by the typical saccade length between them (approximately 2° visual angle [91]) are not guaranteed to always work.

Finding the number of meaningful clusters automatically is an unsolved problem. We ascertained the number of clusters K based on detailed observations of the professionally produced *Watchmen* motion comic DVD from DC Comics. Two independent coders coded the camera moves for panels from four pages in the first chapter. Coder 1 marked 19 with two-point moves and coder 2 marked 24 panels with two-point moves; on average 77% of the camera moves were marked with two points of interest. Other moves marked by the coders were single-point moves when the camera tracked in or out but did not pan significantly, and three or more point moves, when the objects in the panel moved and the camera followed them for example. For moves on stills, two points of interest are largely sufficient. Therefore, we specified $K = 2$ for all our examples. In the k-means algorithm, this value is straightforward to specify.

3.4.2 Pan across the image plane

The *pan* shot is generated by moving the viewport from one point of interest to the other. Let the direction of the pan be represented by the indicator variable $\Theta \in \{0, 1\}$,

$$\Theta = \begin{cases} 0 & \text{if} \quad \mu_0 \rightarrow \mu_1, \\ 1 & \text{if} \quad \mu_1 \rightarrow \mu_0, \end{cases} \tag{3.4}$$

where $\mu_0 \rightarrow \mu_1$ denotes that the start point of interest is μ_0 (the centroid of the first cluster) and the end point of interest is μ_1 (the centroid of the second cluster), and $\mu_1 \rightarrow \mu_0$ denotes the opposite direction. This direction is predicted based on two cues: the onset cluster ϕ and the dominant direction ξ.

The onset cluster is the region that a new observer will likely look at when they first look at the given still image, that is, it is the cluster that contains the onset fixations of the observers. The onset cluster $\phi \in \{0, 1\}$ takes the value $\phi = 0$ if the first and second fixations fall in the cluster 1 and takes the value $\phi = 1$ if the first and second fixations fall in cluster 2. The dominant direction is the direction in which an observer is likely to peruse the image, that is, the more likely saccade direction. The direction indicator $\xi \in \{0, 1\}$ takes the value $\xi = 0$ if the saccade direction is more often from cluster 1 to cluster 2 and $\xi = 0$ otherwise. Because people often look back and forth between two clusters, when the number of saccades from cluster 1 to cluster 2 is equal to the number of saccades from cluster 2 to cluster 1, the onset cluster is the tie-breaker.

The cues are integrated into a likelihood L that the pan will follow the direction represented by Θ,

$$L(\Theta = i) = p(\phi = i)p(\xi = i). \tag{3.5}$$

Let us denote the number of first and second fixations observed in a cluster by $n(\phi = i)$. Then,

$p(\phi = i)$ is computed as

$$n(\phi = i) = \sum_{s=1}^{S} \delta(z_1^{ps} - i) + \sum_{s=1}^{S} \delta(z_2^{ps} - i), \tag{3.6}$$

$$p(\phi = i) = \frac{n(\phi = i)}{\sum_{j=1}^{K} n(\phi = j)}. \tag{3.7}$$

Both the first and second fixations are included while computing ϕ because the first fixation may not be captured reliably, and may be subject to the center bias, that is, the tendency for observers to fixate in the center of the screen when the visual stimulus switches [117].

The probability that the dominant direction of eye movement is $\xi = 0$ is proportional to the number of times people look from the cluster $k_1 = 1$ to the cluster $k_2 = 2$, and vice versa for $\xi = 1$. Let us denote this count by $n(\xi = i)$. Then,

$$n(\xi = i) = \sum_{s=1}^{S} \sum_{l=1}^{\gamma_{ps}-1} \delta(z_l^{ps} - k_1)\delta(z_{l+1}^{ps} - k_2), \tag{3.8}$$

$$p(\xi = i) = \frac{n(\xi = i)}{\sum_{j=0}^{1} n(\xi = j)}. \tag{3.9}$$

We estimate the direction of the pan across the image as

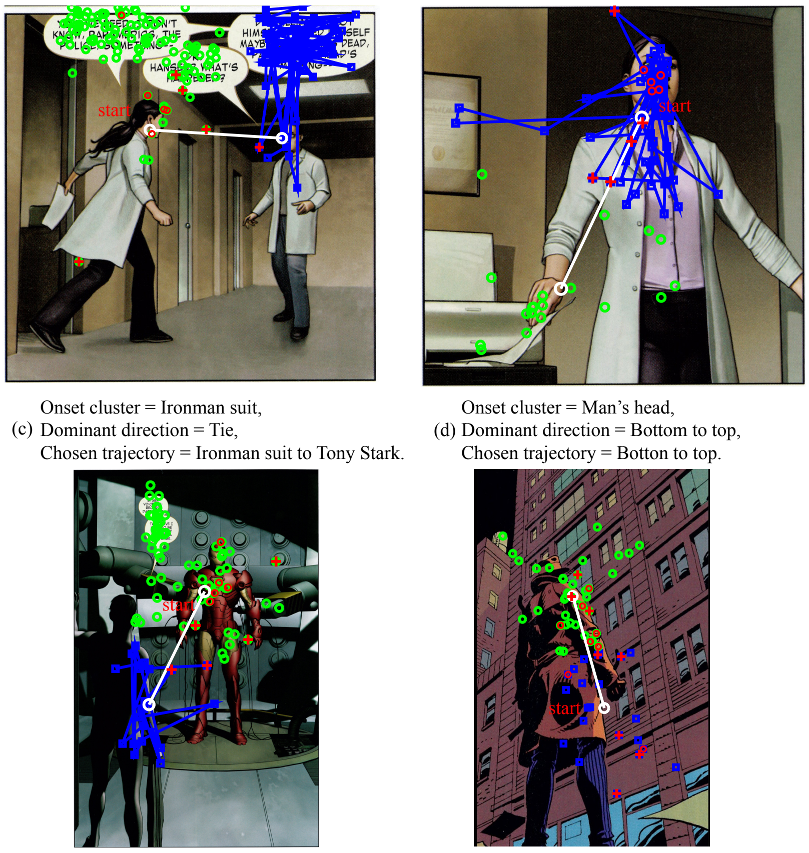
$$\Theta_{MLE} = \arg\max_{\Theta} \quad L(\Theta). \tag{3.10}$$

The examples in Figure 3.8 illustrate the various possibilities for the two cues. In panel (a), ϕ and ξ both vote for cluster 2 as the starting cluster. In panels (b) and (c), there is no clear dominant direction (the observers look back and forth between the clusters) and the onset cluster ϕ is the tie-breaker for the estimated direction of pan. In panel (d), the onset cluster ϕ is the man's head, while the dominant direction ξ is bottom to top. The onset cluster is the man's head because the first fixations were mostly at the bottom part of the screen, but the second fixations were on the man's head. However, the observers all finished looking at the image from bottom to top, leading to a strong dominant direction. The chosen direction for the pan is bottom to top for this panel.

3.4.3 Track in or out

The track in or out shot refers to the movement of the camera towards or away from a subject "...to increase or decrease its importance within the context of a story..." [59]. For a move-on-still, the camera should track in if the end point is more important than the start point and track out if the start point is more important. Because people dwell longer on regions that are more interesting or important to them, the dwell time τ_i on a point of interest μ_i is a natural cue to drive the track shot.

Let μ_i be the start point of interest and μ_j be the end point of interest $(i, j \in \{0, 1\})$. Then, the tracking shot is a track in if $\tau_i < \tau_j$ and track out if $\tau_i > \tau_j$. It is generated digitally by



(a) Onset cluster = Woman's face, Dominant direction = Woman's face to man's face, Chosen trajectory = Woman's face to man's face.

(b) Onset cluster = Woman's face, Dominant direction = Tie, Chosen trajectory = Woman's face to hand.

(c) Onset cluster = Ironman suit, Dominant direction = Tie, Chosen trajectory = Ironman suit to Tony Stark.

(d) Onset cluster = Man's head, Dominant direction = Bottom to top, Chosen trajectory = Botton to top.

Figure 3.8: Fixation data overlaid on four example panels. The green and blue markers are for the different clusters. The red crosses indicate the first fixation for an observer and the red circles indicate the second fixation. The start cluster is marked with a red label.

increasing or decreasing the size of the framing window based on the ratio of the dwell times. For a track in, the size of the start window is the largest window of aspect ratio α that can fit inside the panel and the size of the end window is smaller by the scaling factor λ_j,

$$\lambda_j = c + (1-c)\frac{\tau_i}{\tau_j} \qquad \text{where} \quad \tau_i \leq \tau_j. \tag{3.11}$$

For a track out, the size of the end window is the largest window of the desired aspect ratio α that can fit inside the panel and the size of the start window is smaller by the scaling factor λ_i,

$$\lambda_i = c + (1-c)\frac{\tau_j}{\tau_i} \qquad \text{where} \quad \tau_i > \tau_j. \tag{3.12}$$

The parameter $c = 0.75$ for the *Watchmen* panels, and $c = 0.5$ for all other comic panels. This parameter allows the user to control how sharply the camera will track in or out. For example, a higher value of c would mean that there is not much difference in the sizes of the start and end

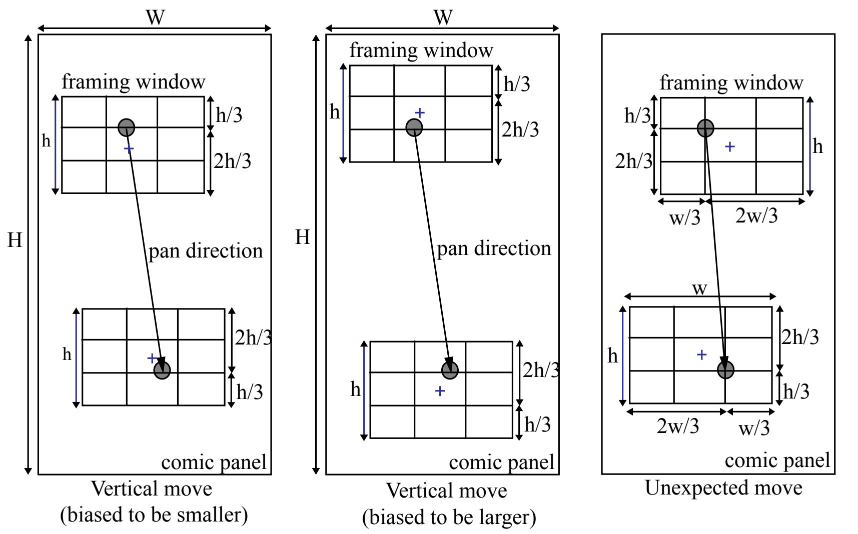


Figure 3.9: Left and middle: the center of the frame is computed so that the point of interest is framed according to the rule of thirds, and such that the point of interest has the same relative spatial location inside the framing window as it does in the panel. Right: we could place the framing window such that the start point of interest lies on the left intersection of the guides and the end point lies on the right intersection of the guides. However, the resulting effect is of a camera moving in the opposite direction because of the selection of guide points.

window, even if the cluster dwell times are unequal. A lower value of c would result in a larger difference in window sizes, to create a more dramatic effect, for example. In our examples, we chose the values of c visually, based on how dramatic a track was comfortable to watch and the typical sizes of faces in each set.

The final dimensions $\mathbf{D}_i$ and $\mathbf{D}_j$ of the framing windows are computed as

$$\mathbf{D}_i = \lambda_i \mathbf{D}, \tag{3.13}$$
$$\mathbf{D}_j = \lambda_j \mathbf{D}, \tag{3.14}$$

where $\mathbf{D}$ denotes the render size, for example, $\mathbf{D} = [640, 480]^T$.The computation thus far fixes the direction the camera will move in a plane parallel to the still image and whether it will move into or away from the still image. The final step in rendering out the move-on-stills is to compute the trajectory of the center of the framing window.

3.4.4 Rendering the move on stills

Because gaze data contains semantic information, we can identify the points of interest in a still picture, the order in which they should be traversed, and their relative importance for the viewer. Gaze data does not, however, provide any information about how the rostrum camera operator would frame these points of interest within the viewport (or framing window). The well-known photography rule of thirds is used to resolve this ambiguity. According to this rule, the framing window should be divided into a 3×3 grid with vertical and horizontal guides; the point of interest should lie on either a vertical or horizontal guide for a visually pleasing effect.

Our method classifies the predicted move-on-still as a North-South move if the pan direction is between $15°$ and $165°$ and as a South-North move if the pan direction is between $-15°$ and $-165°$ (similarly for East-West and West-East). For a North-South move (Figure 3.9, left), the x position of the framing window is computed such that the ratio of the distances of the point of interest to the boundaries is the same in the framing window as in the original panel. Let the start window be parametrized by the top-left corner (x_1, y_1) and the bottom-right corner (x_2, y_2). Its size $\mathbf{D_i}$ was computed in Equation 3.14. Recall that the position of the point of interest is $(\mu_1(1), \mu_1(2))$. Then,

$$x_1 = \mu_1(1) - \frac{\mu_1(1))}{W}\mathbf{D_i}(1), \tag{3.15}$$

$$x_2 = \mu_1(1) + \frac{W - \mu_1(1))}{W}\mathbf{D_i}(1). \tag{3.16}$$

The y position of the framing window is computed such that the point of interest lies on the horizontal third guides. We experimented with two variants: first, using the upper third guide for the start window and lower third guide for the end window (a smaller camera move, as in Figure 3.9 (left)), and second, using the lower third guide for the start window and the upper third guide for the end window (a larger camera move, as in Figure 3.9 (right)). For the first case,

$$y_1 = \mu_1(2) - \frac{1}{3}\mathbf{D_i}(2), \tag{3.17}$$

$$y_2 = \mu_1(2) - \frac{2}{3}\mathbf{D_i}(2). \tag{3.18}$$

The equations for the second case are similar. We found the larger camera move to be the more pleasing option as it displayed a greater amount of the underlying comic art. In Section 3.6, we also numerically compare the two cases with a professional DVD.

Figure 3.9 (right) shows a possible failure case if we rely only on scripted rules based on the direction of movement and the horizontal and vertical guides (frame the start point of interest on the upper left guide intersection and the end point of interest on the lower right guide intersection). The start point of interest is to the left of the end point of interest, but placing them on the guides leads to an opposite camera move. To avoid such cases, we use the relative spatial location of the point of interest in the still picture to place the framing window.

3.5 Results

We present moves on stills for panels taken from a variety of comic books: *Watchmen*, *Ironman: Extremis*, *The Three Musketeers*, *Hulk:WWH*, *Black Panther*, and *The Amazing Spiderman: New Avengers*. In all the result figures, the fixation locations are colored green (marked as circles) and violet (marked as squares) based on their cluster. The red markers mark the first and second fixation points, which are used to compute the onset cluster probability for the pan shot. The large white markers are the points of interest.

Figure 3.10 shows two example panels from the *Spiderman* comic book. In Panel 3, the points of interest are the sneaking man and the sleeping lady, and in Panel 4, the points of interest are

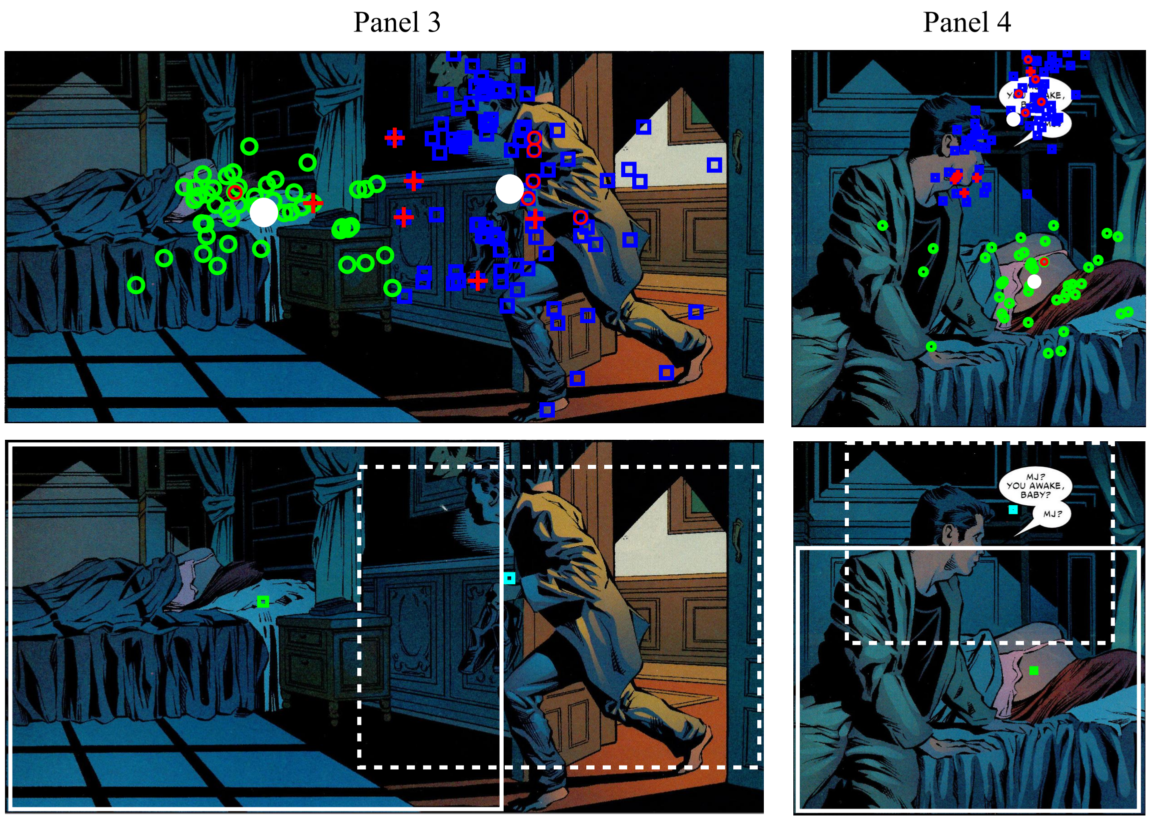


Figure 3.10: Three panels from a *Spiderman* comic book. Top section: fixation locations (green and blue), computed points of interest (white), and the predicted camera move. The dotted rectangle is the start window, the outlined rectangle is the end window. Bottom section: sample frames from the result movie.

the face of the man and the sleeping lady. These panels demonstrate why gaze is a valuable cue—relying on a face detector or similar non-contextual cues would likely have missed the sleeping lady as a point of interest, yet we can see that the artist intended for the viewer to look at her. For Panel 3, the camera panned from right to left and in Panel 4, the camera pans from top to bottom.

Figure 3.11 shows three panels from *Hulk: WWH*. Our algorithm identifies the points of interest as the kneeling man and the cloaked man and predicts a downward pan, a track with a pan, and an upward pan respectively. Sample frames from the result are shown in Figure 3.12. Figure 3.13 shows example panels from the *Ironman* graphic novel. In Panel 3 and Panel 7, the points of interest are clearly the face and hand, and the suit and face, respectively. In Panel 31, the starting point of interest is shifted slightly to the right of the woman's face because of stray fixation points. The camera tracks out for Panels 3 and 7 to reveal the paper in the woman's hand and the man looking at the suit. The camera tracks in towards the man's face for Panel 31, correctly capturing the semantics of the scene.

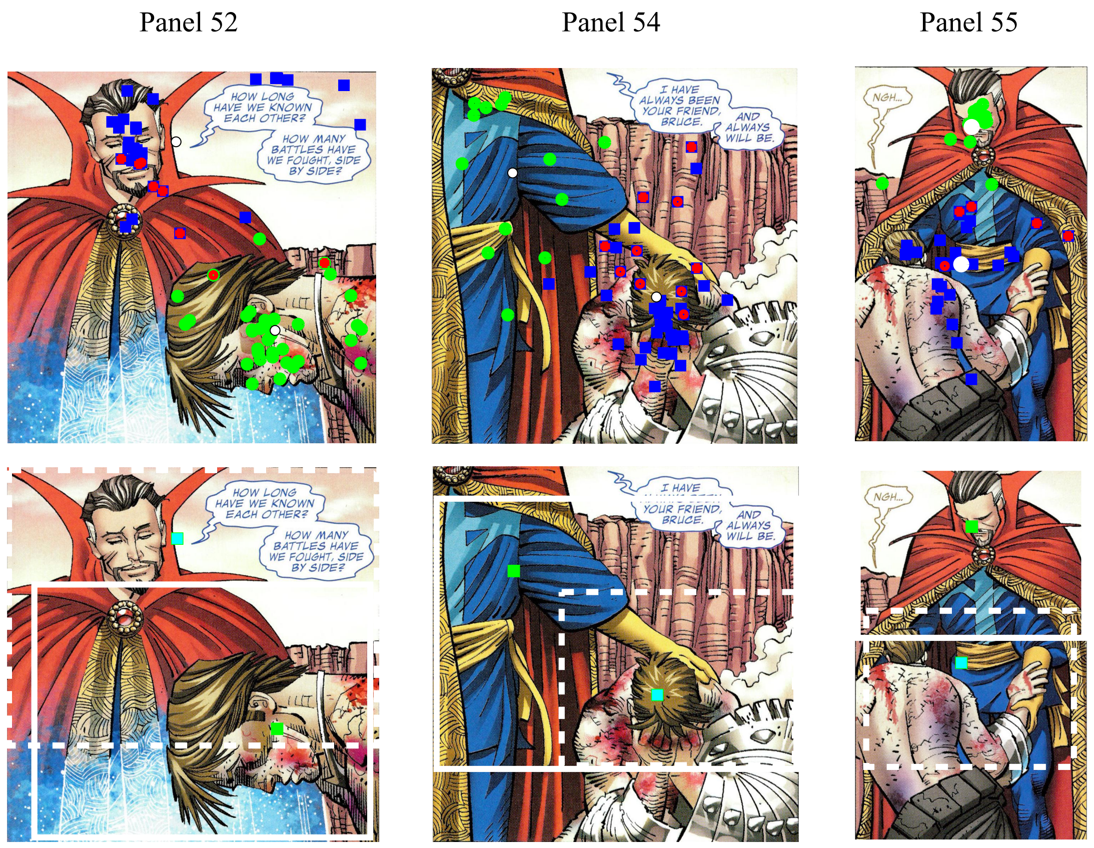

Panel 52
Panel 54
Panel 55
HOW LONG HAVE WE KNOWN EACH OTHER?
HOW MANY BATTLES HAVE WE FOUGHT, SIDE BY SIDE?
I HAVE ALWAYS BEEN YOUR FRIEND, BRUCE.
AND ALWAYS WILL BE.
NGH...

Figure 3.11: Three panels from a *World War Hulk* comic book. Fixation locations (green and blue), computed points of interest (white), and the predicted camera move. The dotted rectangle is the start window, the outlined rectangle is the end window.

3.6 Evaluation

We evaluated the camera moves predicted by our algorithm by comparing with the camera moves created by a professional artist for the *Watchmen* motion comic DVD by DC Comics. Two independent coders watched an excerpt from the DVD and marked the center of the framing window on the corresponding comic panel. The coders coded four comic book pages, comprising 31 panels. Three panels were discarded because a coder made a mistake, or because the panel was not included in the DVD.

Amongst the remaining 28 panels, coder 1 marked 19 with two-point moves and coder 2 marked 24 panels with two-point moves (that is, the camera panned linearly between a start and an end point). Because our algorithm generates two-point moves, we numerically evaluate panels coded with multiple-point moves by considering only the largest segment of camera movement. The average root mean squared distance between the window centers marked by the two coders

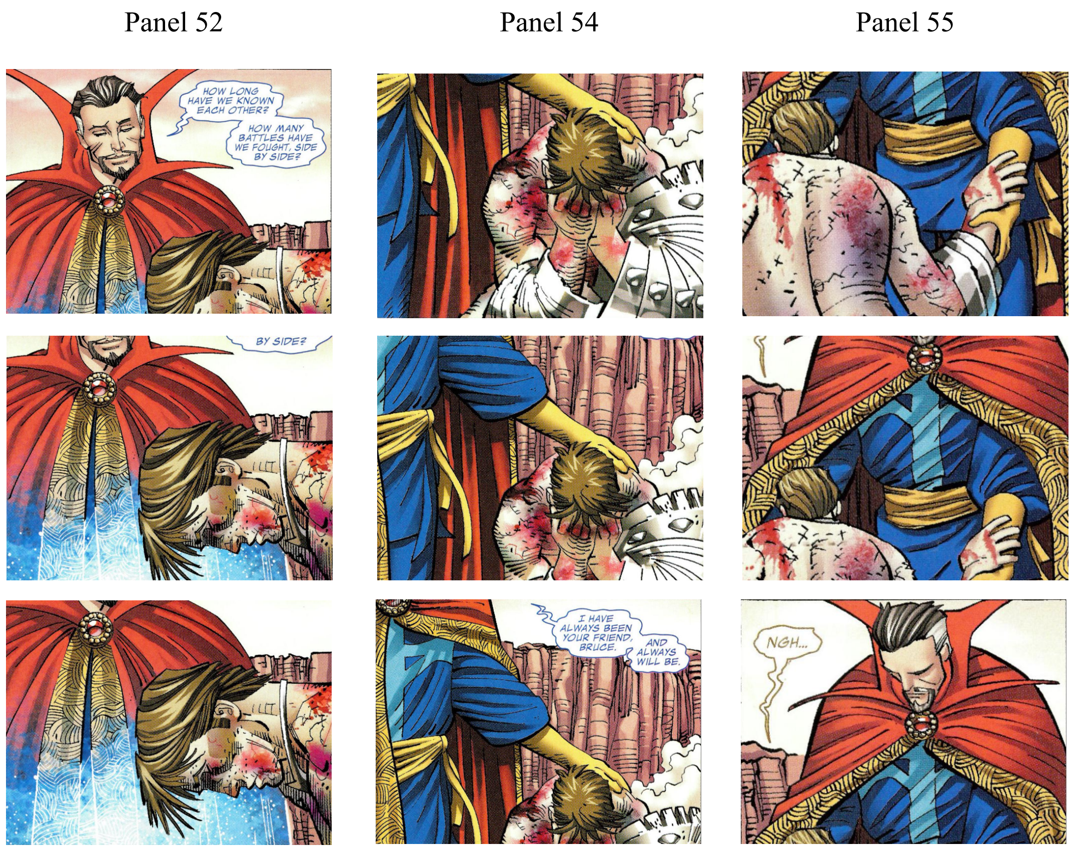

Panel 52
Panel 54
Panel 55
HOW LONG HAVE WE KNOWN EACH OTHER?
HOW MANY BATTLES HAVE WE FOUGHT, SIDE BY SIDE?
BY SIDE?
I HAVE ALWAYS BEEN YOUR FRIEND, BRUCE.
AND ALWAYS WILL BE.
NGH...

Figure 3.12: Three panels from a *World War Hulk* comic book. Sample frames from the result movie.

was 104.8 (in pixels), which is approximately 10% of the largest dimension of a panel. We filtered out the cases where the coders were not consistent by discarding the panels where the distance between their annotations was greater than 10% of the panel dimension (105 pixels). Thus, we have reliable ground truth for 20 panels.

Figure 3.14 (a) is a polar plot where the radial distance is the root mean squared distance between the window center predicted by our method and the window center for a professional DVD, averaged over both coders, and the angular value is the included angle between the direction predicted by our method and the professional DVD. The blue circles mark the version of our algorithm which is biased towards smaller camera moves, while the red squares show the version of our algorithm which is biased towards showing more of the underlying artwork (consequently, a larger camera move). The green triangle markers are a procedural camera move: the camera moves vertically or horizontally based on the aspect ratio of the input comic panel, the direction (whether up or down, left or right) depends on a coin toss. The plotted points were obtained by averaging five runs. An example run is shown with black plus signs.

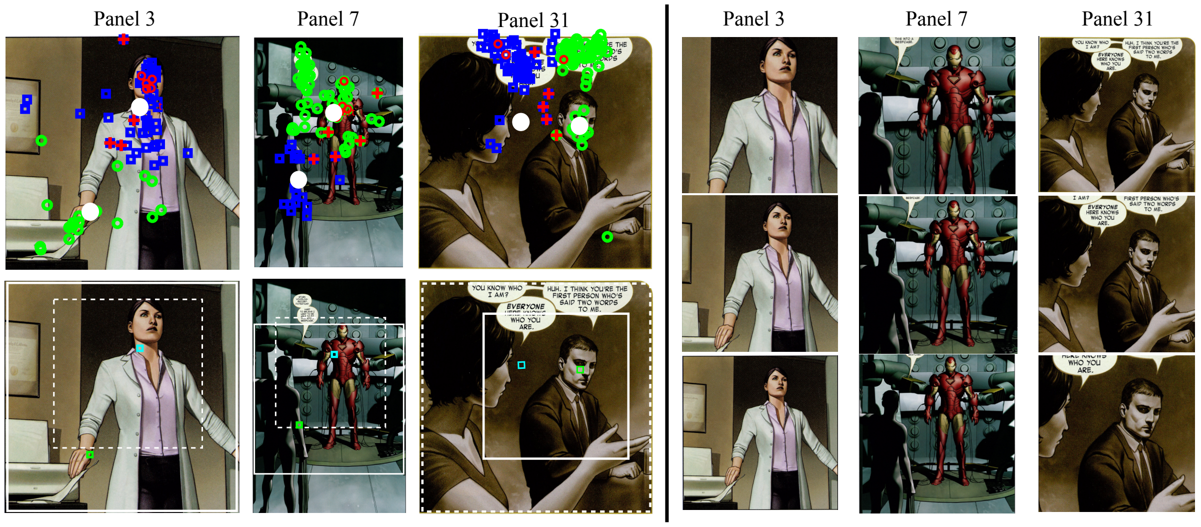

Panel 3
Panel 7
Panel 31
Panel 3
Panel 7
Panel 31
YOU KNOW WHO I AM?
HUH. I THINK YOU'RE THE FIRST PERSON WHO'S SAID TWO WORDS TO ME.
EVERYONE HERE KNOWS WHO YOU ARE.

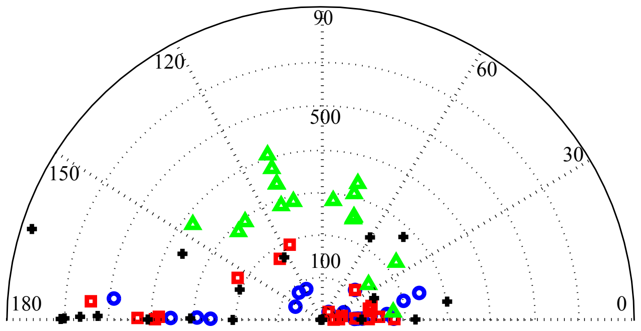
Figure 3.13: Three panels from *Ironman*. Top: fixation locations and the computed points of interest. Bottom: camera move predicted by our algorithm. The dotted rectangle is the start window, the outlined rectangle is the end window.

Points closer to the origin and the $0°$ line represent good performance because the predicted camera parameters match the DVD both in the location of the framing windows and the direction of the move. Points on the $180°$ line show the panels where our algorithm predicted a pan direction opposite to the DVD. Our gaze-driven method predicts the same direction as the DVD in 12 of the 20 test panels. In 2 panels, we predict a pan, whereas the coders annotate the camera move to be an in-place track. For 6 panels, we predict a direction opposite to the DVD. This might occur because people tend to look at faces first; if there is no clear dominant direction, the onset cluster is responsible for the direction of the camera move.

To better understand how well we were able to identify the points of interest (independent of direction), we computed the mean distance by ignoring the order of the start and end centers. Procedure 2 (our method, biased towards showing more of the comic panel) and procedure 1 (our method, smaller move) are equivalent in terms of center distance, averaged over the test panels (115.8 and 115.6), and are closer to the DVD than the aspect-ratio-based move (average distance to DVD = 180). The advantage of our method is especially clear in the panels where the method is able to correctly identify that the points of interest are not located near the top or the bottom of the panel.

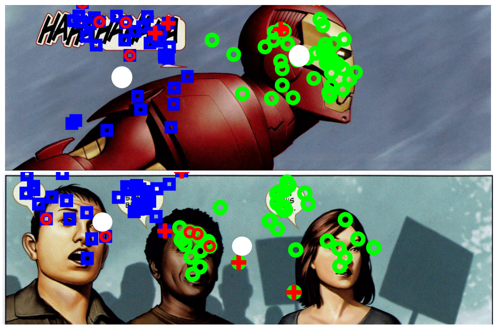
3.7 Discussion

Camera moves are the product of sophisticated creative thought; higher order cognitive processes are involved in recognizing visual elements, inferring the relationship between them, and figuring out a camera move that supports the narrative in the still picture. Though the cognitive processes are not well-understood, a behavioral cue is available to be recorded: gaze. Our key insight is that recorded gaze data contains the information needed to predict plausible moves-on-stills for comic art pictures, if the comic book artist was successful in directing the flow of viewer



○ Procedure 1: smaller move
▣ Procedure 2: bias to edge
▲ Procedure 3: aspect ratio move (average of 5 runs)
✚ Sample run from Procedure 3

(a)



(b)

Figure 3.14: Evaluation and limitations. (a) Evaluation of our method based on ground truth obtained from a professional DVD. We show two versions of our method, along with a naive procedural camera move, shown in green. Points closer to the origin and the $0°$ line represent a good match with the DVD. (b) Top: because people fixate on the back of the head as well as the face, the cluster center shifts towards the ear. Bottom: though the camera move may be a two-point move, two clusters may not accurately extract the points of interest.

attention.

We recorded the eye movements of human participants and found increased similarity across viewers for comic panels, compared with images that were not created with the intent of directing viewer attention (robot pictures, amateur snapshots). We demonstrated an algorithm to extract the parameters of a two-point camera move from the gaze data of viewers over comic panels.

A limitation of our method is that though human fixations are clustered around objects of interest in the image, the computed centroid of fixation locations is a partial representation of this object. For example, in Figure 3.14 (b)(top), people fixate on the back of the head as well as the face, and the k-means algorithm computes the cluster centroid to be closer to the ear than the front of the face. To fully describe the object of interest, we would need to compute a bounding box in addition to the centroid, to ensure that the camera frames the entire face.

Our current implementation also assumes that there are (for the most part) two points of interest in a comic book panel, and thus, it is sufficient to extract two clusters from the recorded viewer fixation points. We observed that even though the camera move may be well specified as a two-point move, the eye movements may not cluster well into two clusters, as in Figure 3.14 (b)(bottom). In this case, an algorithm that continues to pan till it has enclosed all the fixated locations might capture the underlying scene better.

We create the track in or out shot from the ratio of the total fixation durations on the associated clusters because people tend to fixate longer on the region that is more interesting to them; they want to look at this region more closely, therefore a camera should track in towards it. Other cues to predict the size of the window could include metrics that describe the spread of the fixation locations in the associated cluster, such as the distance between the two farthest points in a cluster or the standard deviation of the distribution of fixation locations. Though these metrics could be noisy when the fixation points in a cluster are not well modeled as a Gaussian (for example, the green cluster in Figure 3.8(b)) they probably encode the same information as the fixation durations. It is theorized that there are two modes of visual processing, ambient processing, which is characterized by larger saccade amplitudes and shorter fixation durations (perhaps to explore the spatial arrangement of the scene), and focal processing, which is characterized by longer fixation durations and shorter saccades (probably to gather detailed information about a particular object) [4, 82, 120]. Longer total fixation duration on a cluster would likely be associated with a smaller spread in the fixation locations.

Finally, it may happen that camera moves created by a cinematographer, who knows where the story is going, will contain intentional reveals. Because our algorithm uses information from gaze data on only the current panel (and the viewer does not know what is going to happen next), the predicted moves will contain only as much surprising revelation as the comic book artist was able to achieve through manipulating viewer gaze.

Chapter 4

Retargeting Videos using Pan-and-Scan with Cuts

4.1 Introduction

Viewers consume digital content on a wide variety of display devices, ranging from handheld personal displays, such as iPods or cellphones, to large displays, such as projection systems in movie theaters. When the aspect ratios of production and display are drastically different (e.g., watching films on a smartphone display), a significant modification of the content is required that, as much as possible, still preserves the narrative of the original video. This goal of preserving narrative makes it challenging to develop algorithms that transform content for display at non-native aspect ratios.

Scaling, cropping, and letterboxing have traditionally been used to fit videos to different display devices. In recent years, there has been a growing interest in retargeting videos nonuniformly to different aspect ratios (see, for example, these course notes [107]). However, despite the increasing sophistication of the retargeting operators, most methods rely on computable measures of video saliency and are susceptible to noticeable artifacts. For example, an object with a strong vertical line such as a tree or a pole may shimmer if it is retargeted with an operator that lacks temporal coherence. These approaches mainly concentrate on changing the size of the frame, whereas, a more general solution would re-create the movie for the new display. Because it is usually impossible to reshoot scenes, change the shooting angle, or add new shots, our challenge is to re-edit a movie using only the original video.

In this chapter, we use two principal operations to re-edit a movie while fitting it to a different display size. The first, pan-and-scan, is widely used in commercial adaptations of movies to different aspect ratios. Pan-and-scan is the only nonuniform retargeting operator that guarantees no distortion in the retargeted result, although it does remove parts of the original content. Because the operator can only crop, all scene structure is preserved: lines will remain straight and people will not be made tall and skinny. The second operation, introducing cuts, breaks an original continuous shot into two shots by computing a good location of a new cut inside it [66]. We present a method that optimizes the use of these two operations for re-editing an existing movie. Large changes in aspect ratio between production and display necessitate operators such as cropping and cutting, which make hard commitments as to what is and what is not included in the frame. It is vital, therefore, that the algorithm reliably determine the parts of each frame that are integral to the narrative in the original video.

Viewer gaze data is a measurement of video saliency; this data identifies what people are attending to in the video, i.e., what is really important for the telling of the story in the video. Previous research has established that using information from eyetracking data can improve the performance of computational image saliency algorithms [56]. It has also been shown that even relatively large artifacts in an image remain unnoticed outside the gaze area of the viewer [14]. Though a variety of automatic measures have been proposed for computing visual saliency, they remain inadequate because they do not take into account scene context or storyline, and therefore, easily misfire. Computing saliency for videos is also much more challenging than for still images because additional cues, such as motion and audio, may affect the saliency of objects in the scene. For example, a pedestrian moving across a flowing crowd will immediately become salient because of his opposing motion pattern [102], whereas he might not stand out in a still taken from the video where his facing direction would be the only cue to his unique role. In this chapter, rather than trying to create computational models of visual saliency, we let viewers directly reveal

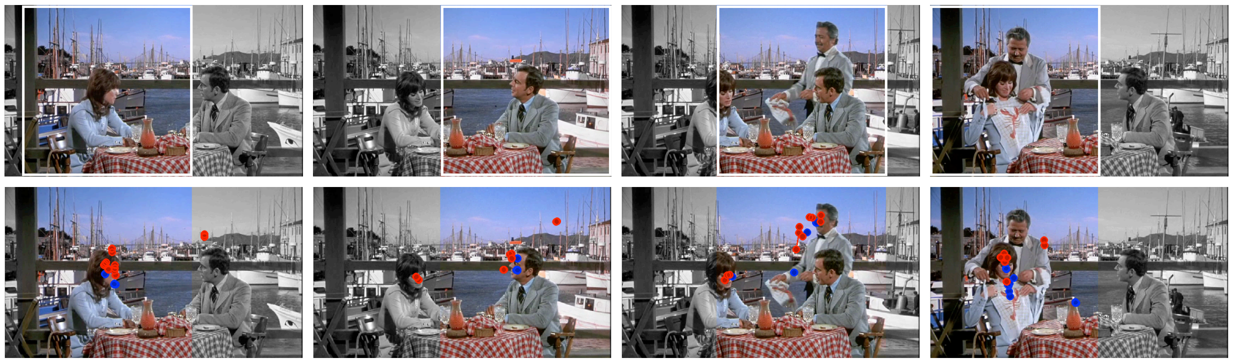


Figure 4.1: We present a gaze-driven algorithm to re-edit widescreen video (1.75:1) to smaller aspect ratios via pan-and-scan with cuts. Top row: The retargeting computed by our algorithm for a 1:1 target aspect ratio is shown in color. The original widescreen frame is in grayscale. The cropping window cuts from the woman to the man and then pans to follow the waiter when he enters the screen on the right and walks over to the left. Bottom row: We record gaze data from viewers watching the original widescreen video (shown in red). Our algorithm uses this data to compute the paths of two cropping windows and an optimal cut between them. We evaluate our results by eyetracking viewers on the 1:1 videos (shown in blue).

what is important to them by eyetracking them while they watch the original video. Thus, viewer attention drives the re-edits created by our algorithm.

In Figure 4.2, we plot the recorded gaze data for the "couple at the waterfront" video shown in Figure 4.1. If the time dimension is flattened out and we look at the x-y plot (overlaid on a sample frame taken from the video), it is not apparent that viewers initially attended to the left portion of the widescreen frame, then shifted to the right and then came back again to the left. This temporal information is important because the underlying visual stimulus is changing with time; when viewers return to the left of the screen, they are watching something different from what they were watching a few seconds ago. In contrast, when viewers look at images, repeated scanpaths to the same portion of the screen indicate that the viewers are watching the

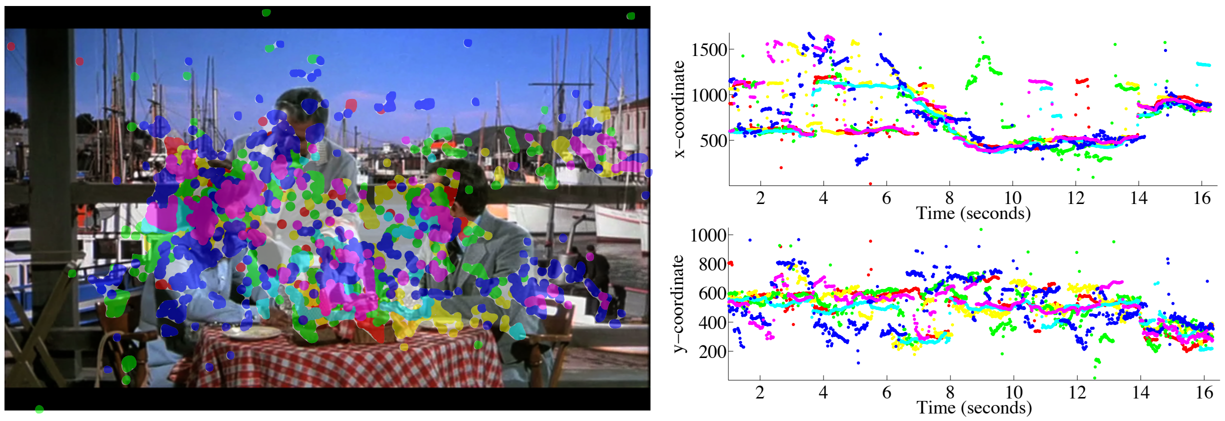


Figure 4.2: We plot the recorded gaze data as an x-y plot, x-t plot and y-t plot.

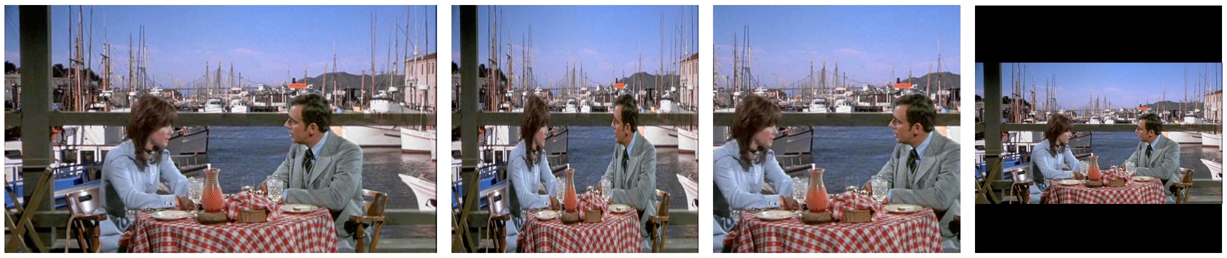

(a) Original frame (1.75:1) (b) Scaling (1:1) (c) Cropping (1:1) (d) Letterboxing (1:1)

Figure 4.3: Commonly used methods to resize video: (a) Original frame, (b) Scaling squeezes the objects and characters, (c) Cropping removes content (d) Letterboxing wastes screen space.

same portion of the stimulus image as they did the previous time they attended to this region. As a result, if we want to extract the regions of interest for a video from the recorded gaze data, the right dimensions to work in are x-t or y-t, not x-y.

We use recorded gaze data from several subjects as input to a RANSAC algorithm that simultaneously finds pan-and-scan paths and cuts while retaining the salient content of the original video. The random search algorithm of RANSAC provides necessary robustness to individual variation in the gaze patterns across subjects and measurement error. To produce a re-edited video, we use RANSAC to search for a path that moves the cropping window through the video cube while maximizing the number of gaze points included. When cuts are required to produce a video with adequate coverage of the eyetracking data, RANSAC simultaneously finds two paths and an optimal cut between them. To close the loop, we evaluate our results based on the principle that faithful re-edits must preserve the viewer gaze patterns observed in the original video. A comparison of viewer eye movements before and after the application of the operators provides a behavioral metric to evaluate the quality of the re-edits produced by our algorithm.

4.2 Background

Most previous methods concentrated on resizing or retargeting the video content to a different format. The three methods used commonly are scaling, cropping and letterboxing. Scaling simply squeezes (or stretches) the video to match the size of the new display device, thereby changing the shapes of objects in the video (Figure 4.3(b)). Cropping trims the original video to the desired size, which often can result in dropping important content (Figure 4.3(c)). Letterboxing places a black matte around the original video when displaying it on a different device, which wastes screen space, especially on small devices (Figure 4.3(d)). These limitations have led to advances in *content-aware* retargeting methods: resizing the original format video non-uniformly so that it fits the new screen size, minimizing the loss and distortion of visually important content.

Two principal classes of methods were introduced for content-aware retargeting. *Discrete methods* remove or insert pixels in the image, minimizing an energy measure until the modified image reaches the target size. These pixels may be removed from the interior of the image (seam carving) [6] or from the edges of the image (content-aware cropping) [16, 66]. The extension to video is created by defining a cuboid of frames in 3D space-time and removing (or replicating) minimal energy 2D manifolds via graph cuts [103]. Content-aware cropping for video data also involves computing optimal pans or zooms [26, 66]. *Continuous methods* apply a warping function to each frame of the original video such that the shapes of important regions are preserved, and distortions (squeezing or stretching) are absorbed by unimportant regions [122, 124, 125]. These two classes of methods differ in how they manipulate the original video. Discrete methods treat the video as a collection of individual pixels, while continuous methods treat the video data as a continuous signal that is sampled according to a function. Still, the only content-based method that guarantees a complete absence of image artifacts is pan-and-scan.

All retargeting methods rely on an underlying saliency map, which guides the algorithm by identifying which regions of the image or video are the most important to preserve. Importance maps are generated by image and video saliency algorithms using cues such as edges, intensity, contrast [108], and optical flow [66, 123, 124]. Krahenbuhl and colleagues additionally allow a user to specify regions which should remain undistorted [61]. Manual hand-drawn binary maps have also been used in [38]. Our use of recorded viewer gaze data provides a more accurate way to ascertain which regions of the video should be preserved. Eyetracking has been used for image retargeting, first by Santella and colleagues [106] for cropping photographs, and more recently, by Castillo and colleagues [14] for providing the saliency map input to nonlinear image retargeting methods. We extend this idea of employing gaze data to video re-editing.

Dorr and colleagues have shown that eye movements are significantly more consistent across viewers for Hollywood action movies compared to movies of natural scenes, such as a video taken at a beach [68]. As eyetracking technologies become cheaper and more easily available (for example, webcam based eyetracking [1, 3], it will become possible to obtain the gold standard for video saliency by simply crowdsourcing viewer gaze data collection. Hence, we use eye-tracking data both for guiding our optimization as well as for validating our results.

4.3 Method

We measure what is important to viewers in a video by recording their gaze as they watch the video. The gaze data is input to a RANSAC algorithm to find a cropping window path through the video cube that encloses maximum saliency while maintaining the specified smoothness characteristics. When saliency shifts sharply from one part of the original widescreen video to another, we cut between two cropping windows. The time location of the cut is optimized within the RANSAC loop, based on the shift in gaze position and the increase in enclosed saliency as a result of the cut. We now describe the eyetracking procedure, the algorithm to fit a cropping window path to the collected gaze data, and the selection of cuts between two cropping window paths.

4.3.1 Data Collection

We selected a variety of clips from three Hollywood films shot in different native formats (2.35:1, 1.82:1, 1.75:1) as the original widescreen videos that would be processed by our algorithm. A categorization of the clips is presented in Figure 4.6. Six naive participants (1 male, 5 females, age ranging from 21 to 44 years) were recruited from among the university community. All

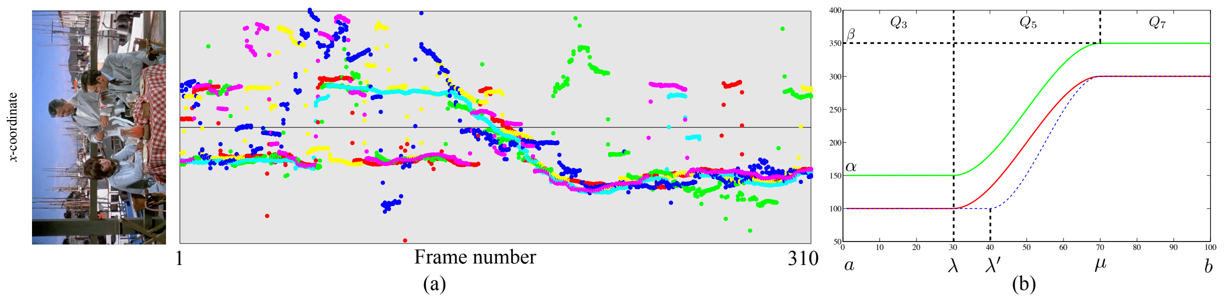


Figure 4.4: Input data and cropping window representation. (a) Viewer gaze is plotted and an example frame is shown from the corresponding video. The x-positions are marked on the vertical axis and frame numbers are on the horizontal axis. Each viewer is assigned a unique color. (b) The path of a cropping window is a piecewise spline curve ($m = 7$) with control points $(\alpha, \alpha, \alpha, \alpha, \beta, \beta, \beta, \beta)$ and knots $(a, a, a, a, \lambda, \lambda, \mu, \mu, b, b, b, b)$. For the curves shown, $a = 1, b = 100$.

participants had normal vision or wore contact lenses or glasses to correct to normal vision and were compensated monetarily for their time.

The video clips were resized to be as large as possible on a 1680×1050 screen, with black letterboxing to preserve the original aspect ratio. Participants were asked to watch these clips. The clips were displayed on a 19 inch screen and the participants sat approximately 18-24 inches from it. A visual angle of 1 degree is approximately 27 pixels at these settings. Before beginning the data collection, participants were asked to adjust the chair to a height and distance comfortable to them; then the system was calibrated. After calibration, they were able to move their head freely while their eyes were tracked. This setup allowed for a natural viewing situation as no chin rest was required. The stimuli video clips were presented in randomized order. Before each clip, a one-line blurb was displayed to provide context. The eye movements of the participants were recorded with SensoMotoric Instruments' RED eyetracker, running the iViewX software at 60Hz. Raw data is illustrated in Figure 4.4 (left).

4.3.2 Cropping window

The pan and scan operator is a cropping window that moves across the original widescreen video. It is parametrized by the position of its center $(x_i, y_i), i = 1, 2, \cdots, F$, where F is the number of frames in the video, and its size $\mathbf{D}_i$, where $\mathbf{D}_i(1) = 1050$ and $\mathbf{D}_i(2) = 1680$, for example. Because we want to show as much of the underlying widescreen frame as possible in the retargeted video, the size $\mathbf{D}_i$ is fixed to be the largest window with the specified aspect ratio that fits in the widescreen video. As a result, the cropping window path is obtained by computing the x-coordinate of the center of the cropping window, i.e., x_i.

A panning camera is often employed to reframe a subject so that it remains in the frame while moving. The pan is thought to mimic the smooth motion of the eye as it follows a subject while keeping the movement minimally noticeable [59]. On the other hand, the purpose of eye movements is to process information, which can occur either by quickly shifting visual attention (saccades), resting at the same location (fixations), or smoothly following a moving target (smooth pursuit). As a result of this difference, naively smoothed or filtered eyetracking data cannot be used to drive the camera's motion. Instead we represent the path of the cropping window with B-spline curves. This representation allows us to enforce the smooth ease-in-ease-out motion that helps keep the pan minimally noticeable. Furthermore, we use a RANSAC algorithm to be robust to noise, which includes both individual variation in eye movements, and measurement noise in the eyetracking device.

Representation

We represent the cropping window path as piecewise B-spline that allows for a flat segment (i.e., stationary cropping window), followed by a smoothly varying segment (i.e., pan with ease-in-ease-out), followed by a flat segment (cropping window stationary again). We impose C^1 smoothness on the computed cropping window path by representing it as a piecewise nonuniform cubic B-spline with repeated knots. A nonuniform cubic B-spline is parametrized by $(m+1)$ control points and $(m+5)$ knots. For $m = 7$, the control points are $\mathbf{P}_0, \mathbf{P}_1, \cdots, \mathbf{P}_7$, and the knots are $t_0, t_1, \cdots, t_{11}$. The first curve segment Q_3 is controlled by $(\mathbf{P}_0, \mathbf{P}_1, \mathbf{P}_2, \mathbf{P}_3)$, and is defined in the interval $[t_3, t_4)$. In general, the curve segment Q_i is computed

$$\begin{aligned} Q_i(t) = & \mathbf{P}_{i-3} \cdot \mathbf{B}_{i-3,4} + \mathbf{P}_{i-2} \cdot \mathbf{B}_{i-2,4} \\ & + \mathbf{P}_{i-1} \cdot \mathbf{B}_{i-1,4} + \mathbf{P}_i \cdot \mathbf{B}_{i,4}, \quad t_i \le t < t_{i+1}, \end{aligned} \tag{4.1}$$

where $\mathbf{B}_{i,j}$ are the blending functions and $i = 3, 4, \cdots, 7$ [36].

By appropriately designing the multiplicity of the knots, we can influence the behavior of the curve segment. We set the control points $(\mathbf{P}_0, \mathbf{P}_1, \mathbf{P}_2, \mathbf{P}_3, \mathbf{P}_4, \mathbf{P}_5, \mathbf{P}_6, \mathbf{P}_7) = (\alpha, \alpha, \alpha, \alpha, \beta, \beta, \beta, \beta)$, and the knots $(t_0, t_1, \cdots, t_{11}) = (a, a, a, a, \lambda, \lambda, \mu, \mu, b, b, b, b)$. With this multiplicity, the curve segment Q_3 is controlled by $(\mathbf{P}_0, \mathbf{P}_1, \mathbf{P}_2, \mathbf{P}_3) = (\alpha, \alpha, \alpha, \alpha)$ in the interval $[t_3, t_4) = [a, \lambda)$. The segment Q_4 is of zero length because $t_4 = t_5 = \lambda$. The segment Q_5 is controlled by $(\mathbf{P}_2, \mathbf{P}_3, \mathbf{P}_4, \mathbf{P}_5) = (\alpha, \alpha, \beta, \beta)$ in the interval $[t_4, t_5) = [\lambda, \mu)$. The segment Q_7 is controlled by $(\mathbf{P}_4, \mathbf{P}_5, \mathbf{P}_6, \mathbf{P}_7) = (\beta, \beta, \beta, \beta)$ in the interval $[t_7, t_8) = [\mu, b)$.

The value a is the start frame of the shot, and the value b is the end frame of the shot. Therefore, the free parameters are $(\alpha, \beta, \lambda, \mu)$. Figure 4.4 illustrates the effect of changing the parameters of the curve. The parameters for the red plot are $\alpha = 100, \beta = 300, \lambda = 30, \mu = 70$. By changing the control points to $\alpha = 150, \beta = 350$, but keeping the same knots, we can shift the curve (green). The blue dotted line shows the effect of changing a knot to $\lambda' = 40$.

Fitting the cropping window path to data

Recorded gaze data consists of noisy measurements of 'what is important' to a viewer because it is subject to individual idiosyncracies and measurement error. We fit a nonuniform cubic B-spline robustly to this data with a RANSAC algorithm [35]. The trial set τ for each RANSAC iteration comprises of four randomly selected gaze points. This selection is done by picking the first four items of a random permutation of all frames, and then selecting a random gaze sample from each frame (to avoid picking two trial points from the same frame).

The goal now is to find a curve, parametrized by $(\alpha, \beta, \lambda, \mu)$, that minimizes the saliency fitting error e_s,

$$e_s = \sum_{i,j \in \tau} ||x_i - \tilde{x}^i_j||_2, \tag{4.2}$$

where x_i is the center of the cropping window for the i^{th} frame (obtained by sampling the curve $Q(t)$ at $t = i$), and $\tilde{x}^i_j$ is the j^{th} recorded gaze point for the i^{th} frame.

Because the knot sequence must be nondecreasing ($a \leq \lambda < \mu \leq b$), there are three linear constraints to be satisfied:

$$a - \lambda \quad \leq 0, \tag{4.3}$$

$$\lambda - \mu \quad \leq K_1, \tag{4.4}$$

$$\mu \quad \leq b, \tag{4.5}$$

where K_1 is a minimum distance between knots. Additionally, the cropping window should not exceed the boundary of the original screen, i.e., $1 + \mathbf{D}_i(2)/2 \leq \alpha, \beta \leq \mathbf{D} - \mathbf{D}_i(2)/2$.

The selection of trial points is repeated N times, and for each trial, we use MATLAB's constrained minimization solver to compute the piecewise spline curve that fits the current trial samples τ. The consensus set for this curve is

$$\eta = \left\{ \tilde{x}^i_j : |x_i - \tilde{x}^i_j| < K_2 \mathbf{D}_i(2) \right\}. \tag{4.6}$$

The score of the associated trial is $|\eta|$. The parameter K_2 is set to $1/3$ to compute the number of gaze points enclosed within the third guides of the cropping window. The curve with the highest score is selected as the cropping window path. The result of this algorithm for an example video is shown in Figure 4.5.

4.3.3 Cuts

When viewer attention shifts quickly across the original widescreen video, a cut may be preferable to a fast-moving cropping window; our method allows the introduction of a cut based on recorded gaze data. We fit two spline curves ${}^A x$ and ${}^B x$ to randomly selected trial sets τ_A and τ_B [1]. A cut is generated by switching from ${}^A x$ to ${}^B x$ based on viewer attention shifts. In practice, we find that cutting more than once in a shot from a professionally edited movie is unnecessary because the shots are tightly edited to begin with.

A shift in viewer attention is computed from the change in median gaze position across consecutive frames. Candidates for cuts are posited when the shift in the median is above a threshold value and the two cropping windows *A* and *B* are sufficiently far apart (to avoid a 'jump' cut, a cut that appears jarring to the viewer because the scenes before and after the cut are

[1] For smarter sampling, τ_B is selected from among the samples outside the consensus set of x^A.

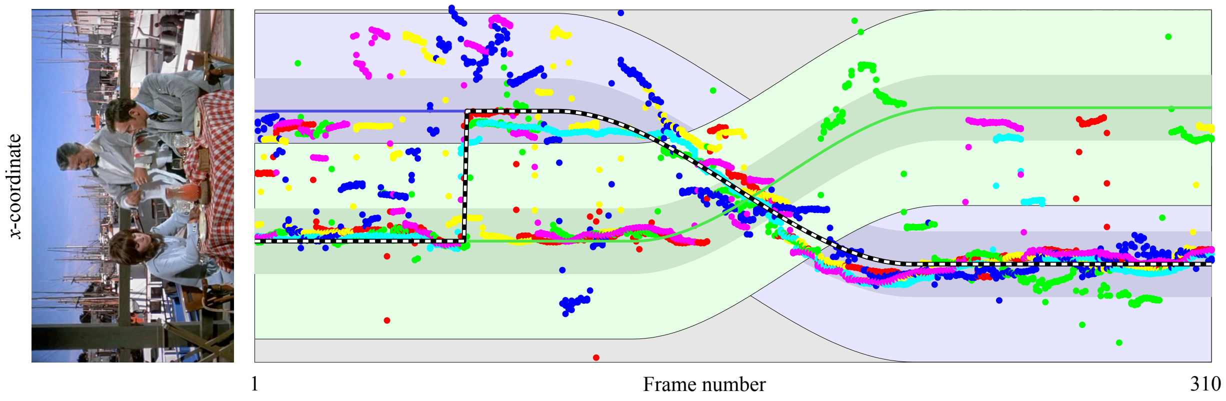


Figure 4.5: The x-coordinate of the recorded gaze data is plotted for each frame of the 'couple at the waterfront' video shown in Figure 4.1. The green and purple curves are the nonuniform B-splines computed by our algorithm corresponding to the two cropping window paths. The black dotted line is the resulting cropping window path after a cut is introduced. The gray band is the original wide screen, the light green and purple bands are the cropping windows at the given aspect ratio (1:1), and the darker green and purple bands represent the region used to compute the consensus set for each RANSAC trial.

too similar [28]):

$$\begin{aligned} \underset{j=1\cdots\gamma_i}{\text{median}}(\tilde{x}_j^{i+1}) - \underset{j=1\cdots\gamma_i}{\text{median}}(\tilde{x}_j^i) &\quad > K_3, & (4.7)\\ |^A x^i -^B x^i| &\quad > K_4, & (4.8)\end{aligned}$$

where γ_i is the number of gaze points recorded for frame i. Each candidate cut κ is ranked by the number of gaze points enclosed by the resulting cropping window path. Let x_i^{final} denote the cropping window path for the cut κ:

$$x_i^{\text{final}} = \begin{cases} {}^A x_i & \text{if } i \leq \kappa, \\ {}^B x_i & \text{otherwise.} \end{cases} \tag{4.9}$$

Then, the consensus set is computed as in Equation 4.6, $\eta = \left\{\tilde{x}_j^i : |x_i^{\text{final}} - \tilde{x}_j^i| < K_2 \mathbf{D}_i(1)\right\}$ and the score of the associated cut is $|\eta|$. If the candidate cut with the highest score κ^* encloses more gaze samples than either one of the individual cropping windows, the corresponding resulting path $x^{\text{final}*}$ is the path selected for this RANSAC iteration.

4.4 Results

We present results on eighteen video clips. Figure 4.6 shows the number of clips in each category. Fourteen clips involve a stationary scene camera, and eight clips involve a moving camera. The foreground objects and background objects both show significant movement in seven clips. The foreground objects are moving while the background is stationary in ten clips. In two clips, both the foreground and background objects show very slight motion.

Background objects are moving.

Scene camera \ Foreground object	slight motion	significant motion
stationary	2 clips	3 clips
moving		4 clips

Background objects are stationary.

Scene camera \ Foreground object	slight motion	significant motion
stationary	2 clips	7 clips
moving	1 clip	3 clips

Figure 4.6: We present results on a variety of clips (12-24 seconds in duration) taken from three Hollywood films: *Herbie Rides Again*, *Who Framed Roger Rabbit*, and *The Black Hole*. The clips are categorized based on the motion of the foreground object, the scene camera and the background objects. When one clip fits more than one category, it is counted in both categories (for example, if the scene camera was stationary for half the duration and moved in the other half).

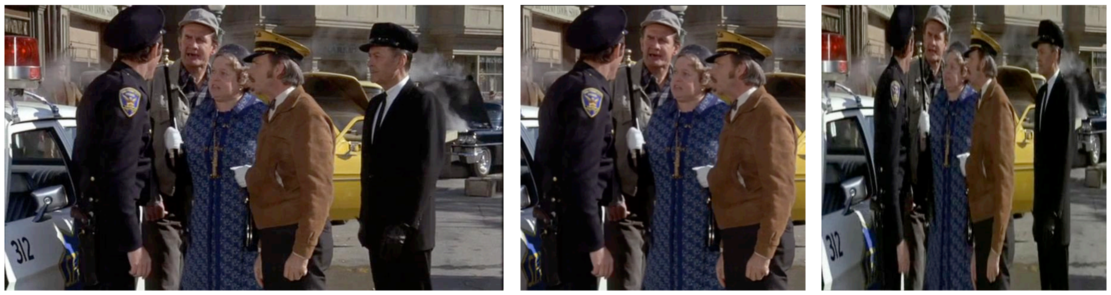
For all examples, the number of RANSAC trials $N = 1000$, $K_1 = 100$, $K_2 = 1/3$, $K_3 = K_1$, and K_4 is set to allow 20% overlap between the two cropping windows *A* and *B*,

$$K_4 = \min(\mathbf{D}_i(2), \max(\mathbf{D}(2) - 1.2\mathbf{D}_i(2), \mathbf{D}_i(2)/2)).$$

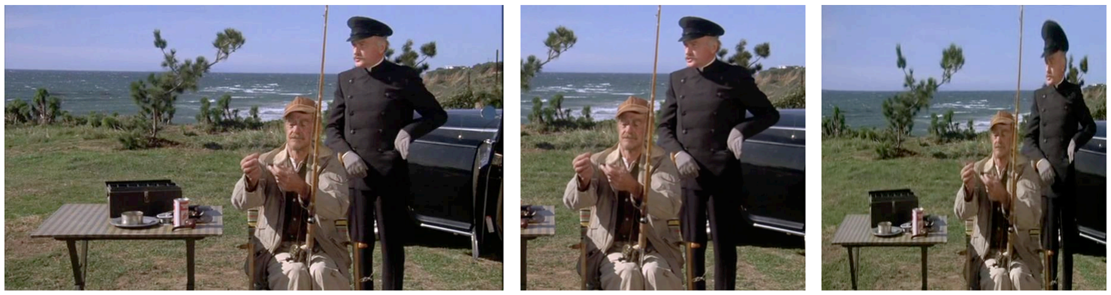
These parameters were selected based on a clip taken from each of the example videos; the algorithm was then run on the entire durations of the example videos and the results are presented in the accompanying video submission. In Figure 4.7, we show a frame each from three example videos at the 1:1 aspect ratio, and in Figure 4.8, we show a result at the 1.3:1 aspect ratio. We also show the result of our algorithm on a sequence provided by Liu and Gleicher [66] in Figure 4.9. A 30 second sequence takes approximately 40 minutes of computation time because the nonuniform B-spline blending functions need to be computed for each RANSAC trial (which could be sped up). We compare our results to the method of Wang et al. [125] and Krahenbuhl et al. [61], which are state-of-the-art methods that run a nonlinear warping operator on the original video. They compute a saliency map based on video features such as edges, contrast and optical flow. Because these methods attempt to preserve important regions while deforming unimportant regions (where importance of a region is based on their computational saliency maps), we observe significant distortion artifacts, such as people being squeezed thin and the shape of the cap changing in Figure 4.7 (middle row). In the video results, we additionally see waving artifacts. Our method, on the other hand, removes content from the scene instead of distorting it (for example, the fisherman's table in the middle row of Figure 4.7. This removal is driven by whether or not viewers attended to that region, therefore, our method is less likely to remove regions that were important to the telling of the story in the video clip.

4.5 Viewer Gaze as a Quality Metric

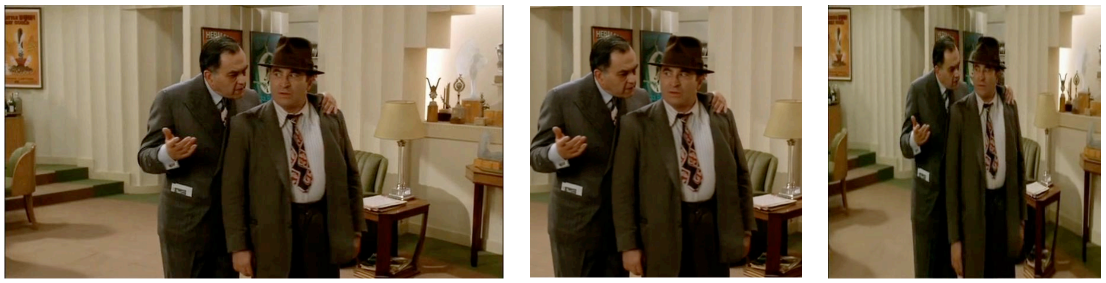
Our method uses viewer gaze to re-edit widescreen video to fit a smaller aspect ratio. In this section, we investigate whether the re-edits performed by our method alter the flow of viewer



Original widescreen video (1.75:1) Our result (1:1) Wang et al. 2011(1:1)



Original widescreen video (1.75:1) Our result (1:1) Wang et al. 2011(1:1)



Original widescreen video (1.83:1) Our result (1:1) Wang et al. 2011(1:1)

Figure 4.7: Sample frames are shown from three example sequences. We compare with the nonuniform crop and warp method of Wang et al(2011). Significant distortions are introduced by their method at large scale changes.

gaze. We run all the evaluation tests on the most aggressive aspect ratio we have, i.e., 1:1. Because our premise is that viewer gaze data is the gold standard for the important regions of the video (the regions that a good video re-editing method should preserve), the first evaluation we perform is the percentage of recorded gaze data on the original video that was included in the result. The included set for each frame i of a video v is

$$\eta'_{iv} = \left\{ \tilde{x}^i_j : |x_i - \tilde{x}^i_j| < \mathbf{D}_i(2) \right\} \quad i = 1, \cdots, N_v, \tag{4.10}$$

where N_v is the total number of frames in the video v. The percentage of gaze data included for a whole video is the mean over all frames of the video and the average percentage for V videos is

$$\eta'' = \frac{1}{V} \sum_{v=1}^{V} \sum_{i=1}^{N_v} \frac{|\eta'_{iv}|}{\gamma_i}. \tag{4.11}$$

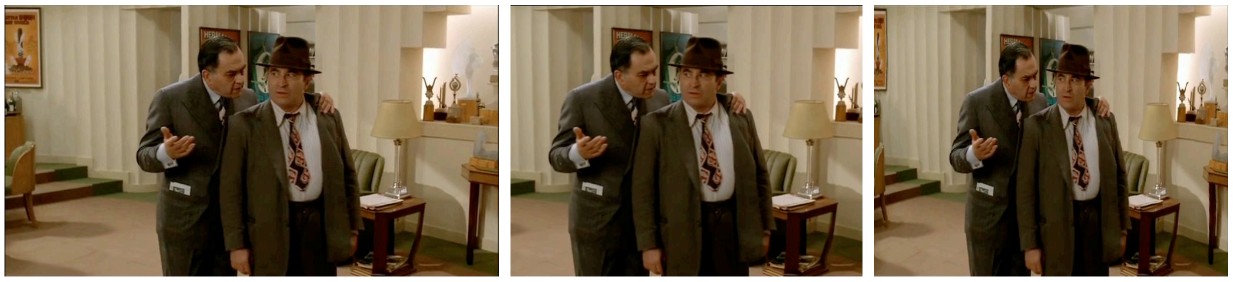

Original widescreen video (1.83:1)
Our result (1.3:1)
Krahenbuhl et al. 2009 (1.3:1)

Figure 4.8: We compare our result at the aspect ratio 1.33:1 with the method of Krahenbuhl et al. (2009), run with the default parameters. Because gaze data informs our algorithm that the two people are important to the viewers while the edges of the frame are not attended to, we are able to crop away the edges of the frame. The other method squeezes the important content.

For the eighteen example clips we tested ($12-43$ seconds each), the percentage of included gaze samples is shown in Figure 4.10. The average included percentage is $\eta'' = 81.9\%$ ($\sigma = 7.4\%$). Figure 4.10 (left) shows the mean included percentage for each example clip as red circular markers. The black plus markers mark the included percentage for a naive re-editing operator: a stationary cropping window of the desired aspect ratio placed at the center of the widescreen video. The percentage of gaze data included in the result of this naive operator is comparable to the included percentage by our method (mean percentage is $81.6\%, \sigma = 9.7\%$) . Figure 4.10 (right) illustrates why this happens. The thin gray dotted lines are the vertical third guides for a sample widescreen frame (1.75:1) and the thick gray dotted lines are the boundary lines for a 1:1 cropping window placed at the center of the widescreen video (the naive operator). For the third guides to be outside the centrally placed cropping window, we need $(w/6) > (h/2)$ i.e. $w > 3h$, which is an atypically wide aspect ratio. As a result, the naive operator yields a high percentage of included gaze samples (because most action occurs within or near the third guides). Even so, Figure 4.11 shows how a centrally placed cropping window may easily fail to capture the important regions of a widescreen video and our method is able to handle such cases.

The second evaluation examines the extent to which our re-editing algorithm alters viewer eye movements. We compare the eyetracking data of viewers watching our result videos with the eyetracking data collected on the original clips. The human participants recruited to watch the result videos were different from the participants who watched the original videos. Thus, we compare first-time viewers for each condition. We compute the distance between the recorded gaze data on the result with the gaze data on the original that is included within the boundary of the result video. Because the videos are now different sizes, we cannot directly compute differences in gaze locations (viewers could be looking at the same object in the video but the recorded data would have a different numeric value). We therefore transform the pixel locations back to the coordinates of the original video. Let w be the inverse operator that transforms the retargeted video to the original video. Then, the median gaze position for each frame i in video

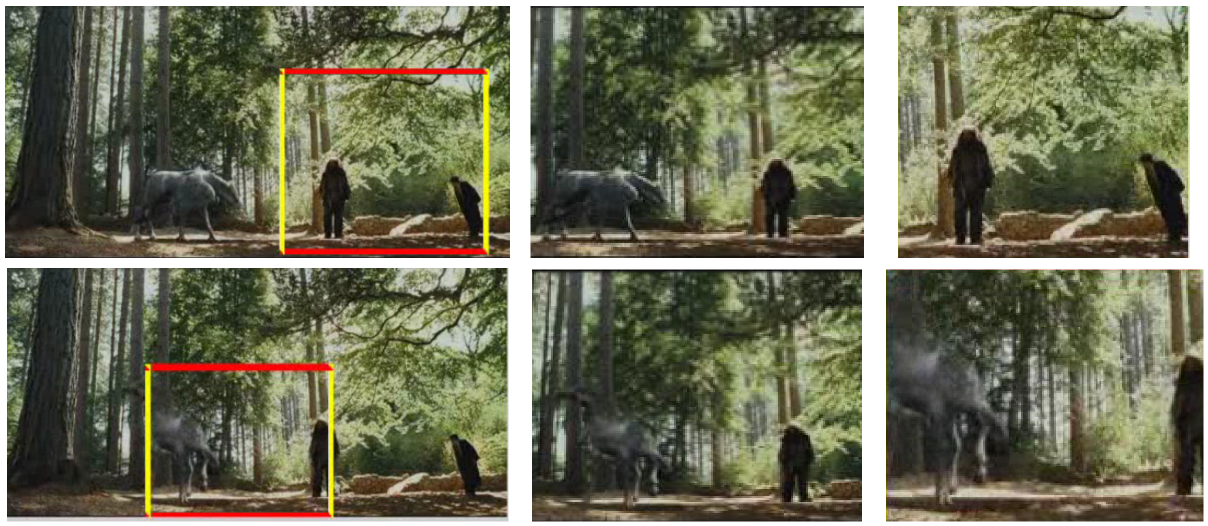

Screenshots taken from the result video sent to us by the authors
Our result (1.3:1)
Liu and Gleicher (2006)

Figure 4.9: We compare our result at the aspect ratio 1.33:1 with the method of Liu and Gleicher (2006). Our algorithm shows the strange horse instead of the person on the side because people tend to look at the bearded man in the center and the horse more than the bowing person on the side. In contrast, their method chooses to cut from the bowing person to the horse. Because their method either pans or cuts, but not both, it crops away the horse's head as it leaps. Our method gives the leaping horse sufficient head room.

v is denoted $\mathbf{r}_v^i$ for the data on the original video and $\mathbf{r'}_v^i$ for the data on the result video.

$$\mathbf{r}_v^i = \underset{j=1\cdots\gamma_{iv}}{\text{median}}([{}^{\text{incl}}\tilde{x}_j^{iv}, {}^{\text{incl}}\,\tilde{y}_j^{iv}]), \tag{4.12}$$

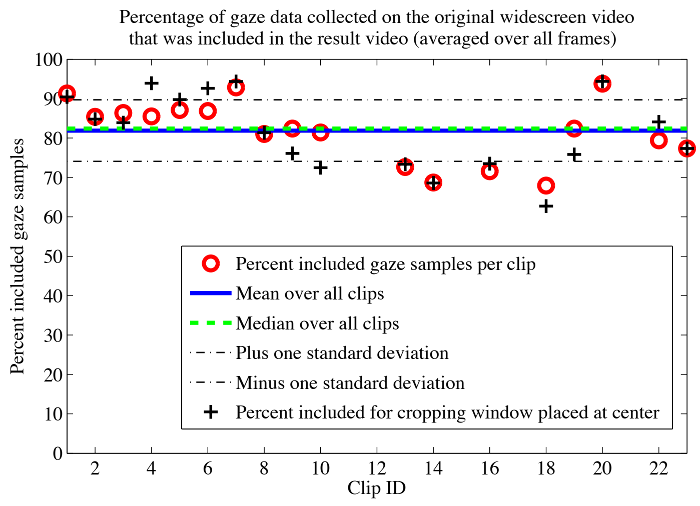
$$\mathbf{r'}_v^i = \underset{j=1\cdots\gamma_{iv}}{\text{median}}(w([\tilde{x'}_j^{iv}, \tilde{y'}_j^{iv}]), \tag{4.13}$$

where $[{}^{\text{incl}}\tilde{x}_j^{iv}, {}^{\text{incl}}\,\tilde{y}_j^{iv}]$ are the gaze samples within the boundary of the result video and γ_{iv} is the total number of gaze samples. Then, the distance δ between included gaze data on the original video (i.e., those red markers that are inside the colored portion of the frame in Figure 4.14) and gaze data on the retargeted video is the mean distance between the median gaze position per frame, averaged over all the frames for video v,

$$\delta = \frac{1}{V}\sum_{v=1}^{V}\frac{1}{N_v}\sum_{i=1}^{N_v}||\mathbf{r'}_v^i - \mathbf{r}_v^i||_2. \tag{4.14}$$

For our eighteen example clips, the distance δ is plotted in red in Figure 4.5 and the average distance $\delta = 79.3$, shown as a blue dotted line. Sample frames from the 'lawyer' video are shown in Figure 4.14 (right).

In Figure 4.14 (left), we plot the median gaze positions for each frame of the 'lawyer' video. The thick blue line represents the median gaze data recorded on our result videos and the thick



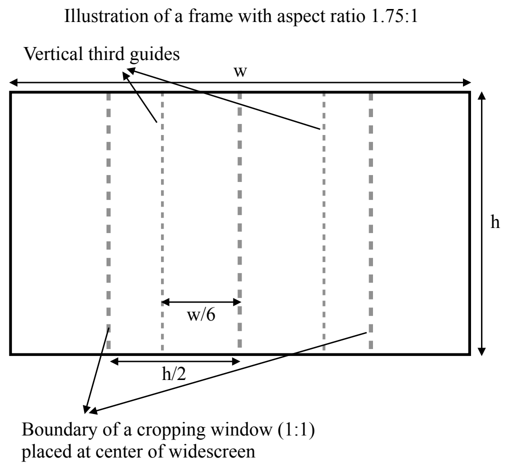


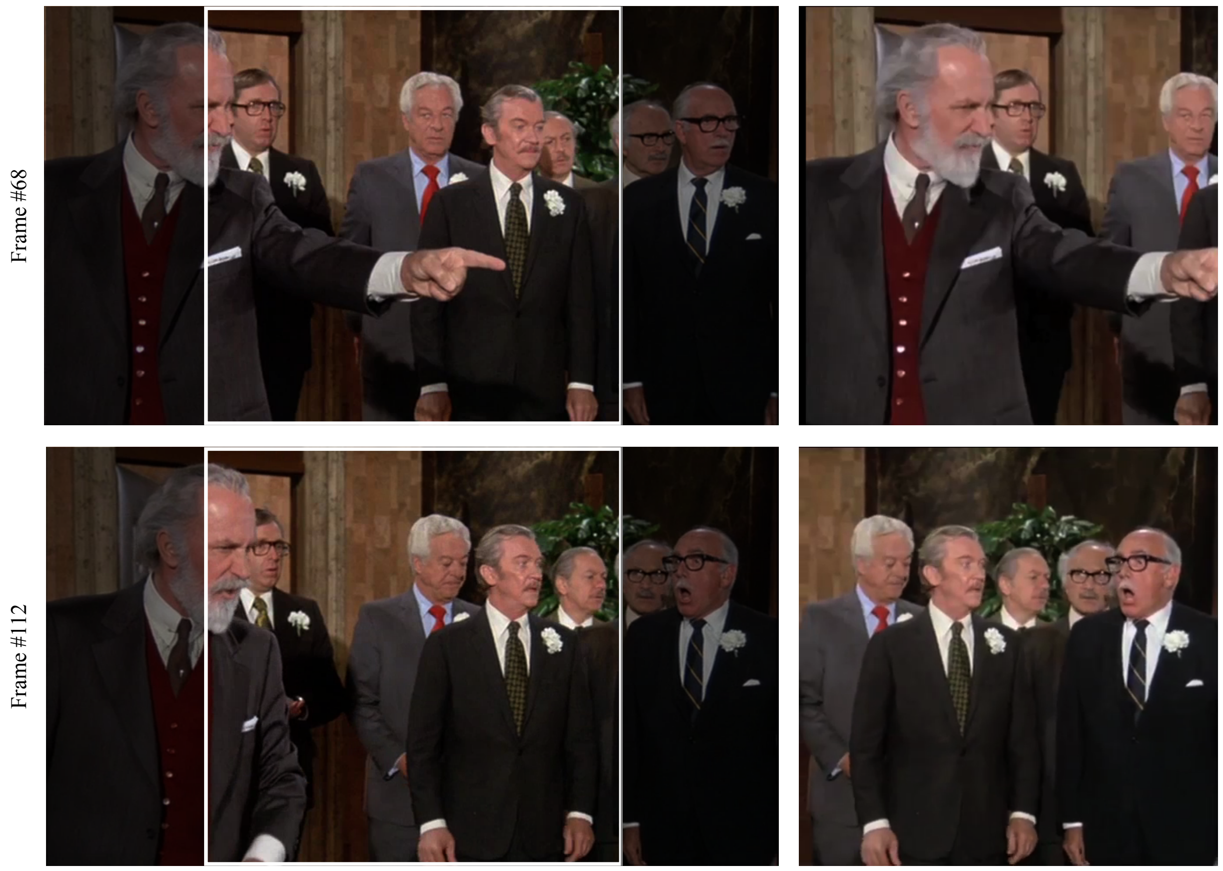
Illustration of a frame with aspect ratio 1.75:1
Vertical third guides
w
h
w/6
h/2
Boundary of a cropping window (1:1) placed at center of widescreen

Figure 4.10: Left: The red markers show the mean percent included gaze samples for each example clip. The mean percentage for all example videos is $81.9\% (\sigma = 7.4\%)$ and the median is 82.4%. The black plus markers show the percent included gaze samples for a cropping window placed at the center of the widescreen. Right: We illustrate why the naive operator (cropping window placed at the center) reports a comparable percentage of included gaze samples to our method. Because most action in a scene usually occurs within or near the third guides of the frame, a large percentage of viewer gaze falls inside the boundary of the centrally placed cropping window when $w < 3h$.

red line represents the median gaze data for the original widescreen videos. The thin blue and red lines represent the standard deviations for each frame. The standard deviation of the gaze data on the original widescreen video is generally larger than the result video because there is more information in the widescreen format. We take a closer look at two cuts introduced by our method; the corresponding frames are shown in the first and third rows in Figure 4.14 (right). Our method selected the appropriate locations to introduce the cuts because viewer gaze shifts from left to right and right to left respectively in the original widescreen video. This shift can be seen in the sample frames and on the graph where we have marked the frames with the label ‘New cut’. The sample frames shown in the second row correspond to a cut that was part of the original video (Frames #333 and #334). It takes some time for the viewers on the original widescreen video to shift their attention to the group of lawyers from the right side of the screen. The phenomenon that the human visual system requires a finite amount of time to respond to a change in stimulus is known as visual persistence [81]. Because our algorithm placed the cropping window to the right side of the original widescreen, the computed distance between gaze data on the original and our result is high for Frame #334.

4.6 Conclusion and Discussion

We have presented a gaze-driven algorithm for re-editing videos to better fit to a smaller aspect ratio. Our method uses the gaze data of viewers on the original video as a means to define the



(a) Centrally placed cropping window (1:1) (b) Our result (1:1)

Figure 4.11: (a) A cropping window placed at the center of the original widescreen frame crops away the bearded man and the talking lawyer. (b) The result produced by our algorithm correctly captures the bearded man and the talking lawyer.

salient parts of the frame. We compute piecewise nonuniform B-spline paths for two cropping windows and find an optimal cut between them using a RANSAC approach. The final results are synthesized as pan-and-scan with cuts between the windows. Finally, we also use eyetracking to evaluate the result videos by comparing the gaze data of viewers on the re-edited videos with the data recorded on the original videos. We found that we succeed in preserving most of the important content of the video in our results.

The vast majority of the eighteen examples that we have run are successful at reducing the size of the image from 2.35:1 to 1:1, 1.3:1 or 1.5:1 without distorting the content or the higher level context of the scene. When the algorithm fails, it is most often because the reduction that we have asked for is impossible without two or more cuts or a high velocity pan. The majority of the original gaze locations are contained within the new video and the new gaze patterns are largely unchanged statistically from those seen in the original.

Unlike the recent nonlinear rescaling algorithms which use seam carving, warping and similar algorithms to "nibble" away at the image, our approach is actually most effective when significant changes are required to the format of the video. This property of our algorithm follows directly

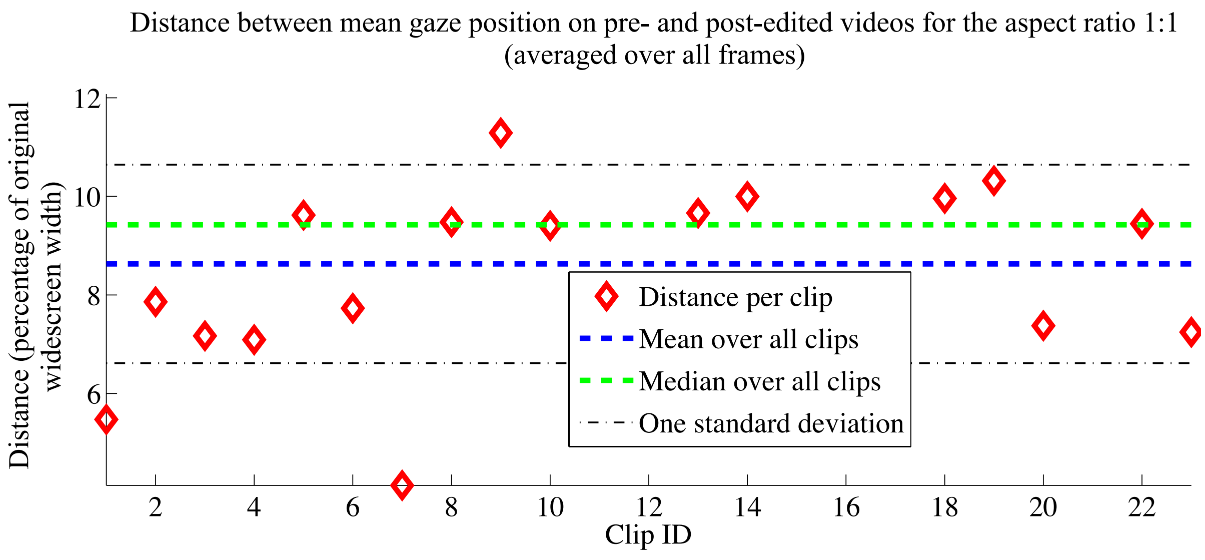


Figure 4.12: The red rhombus markers show the distance between the median gaze position on our results at the 1:1 aspect ratio and the median position of the included gaze samples for the original widescreen video, i.e., the distance between the median of the blue markers and the median of those red markers that lie on the colored portion of the frame in Figure 4.14. The average distance over all example clips is 79.3 pixels ($\sigma = 21.8$).

from our use of gaze data because the gaze pattern clearly identifies what is most salient in the frame but does not always tell us what the lower ranked but still salient objects are unless one or more of the subjects happened to spend time looking at those objects. There are several potential ways to address this issue. We could combine our gaze data with the saliency measures that are available in the literature, giving priority to the 'gold standard' of gaze but using the saliency algorithms to identify other areas of secondary importance. Alternatively, we could use eyetracking on a much larger pool of subjects and use the distribution of their gaze patterns to identify the secondary objects that should also be preserved.

As eyetracking technologies become cheaper and more easily available (for example, webcam-based eyetracking [1, 3]), it will become possible to obtain eyetracking information even by crowdsourcing viewer gaze. Hence, using gaze data with large subject pools will become feasible in terms of both quality and cost. In the future, eyetracking could also perhaps be done on smart personal devices such as smartphones, and on a per user basis to automatically create an personalized version of a movie.

Our algorithm for placing cuts is largely successful in inserting them so that they are not disruptive to the viewing pattern. However, our approach does tend to have a small bias toward using cuts because two windows will always provide more coverage of the salient regions than one, particularly with the relatively slow panning velocities that we have allowed. A statistical analysis of the camera motion and editing patterns used in the original film might help to provide us with additional information about when difficult scenes should be re-edited with more cuts, faster pans, or zooms. [66]. Combining such an analysis with gaze data might also benefit other applications such as reframing for aesthetic reasons (see e.g. [67]) and video summarization.

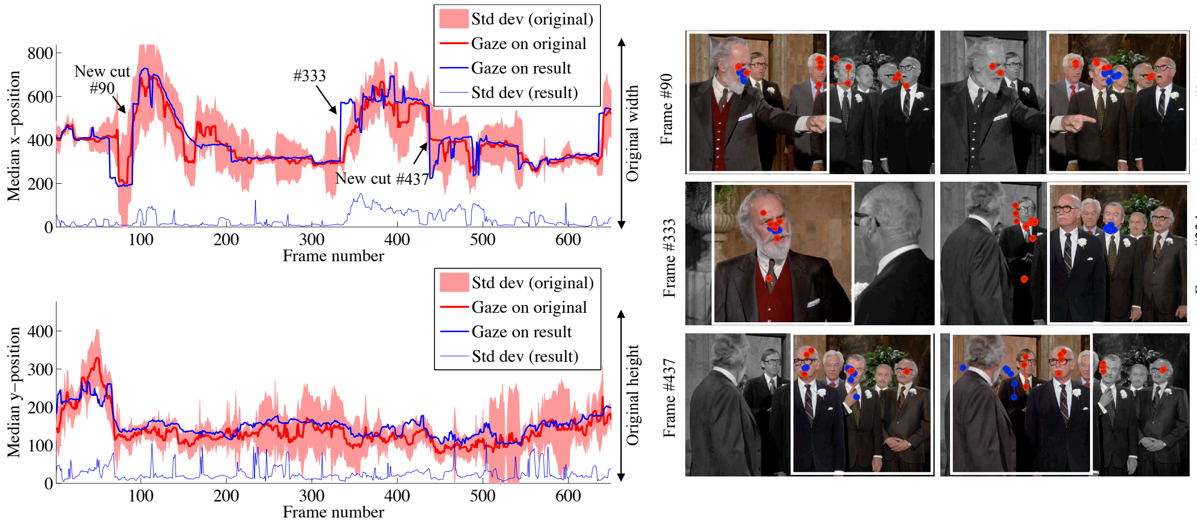


Figure 4.13: The gaze data captured on the original widescreen video is shown in red and the data captured on our result (transformed to widescreen coordinates) is shown in blue.

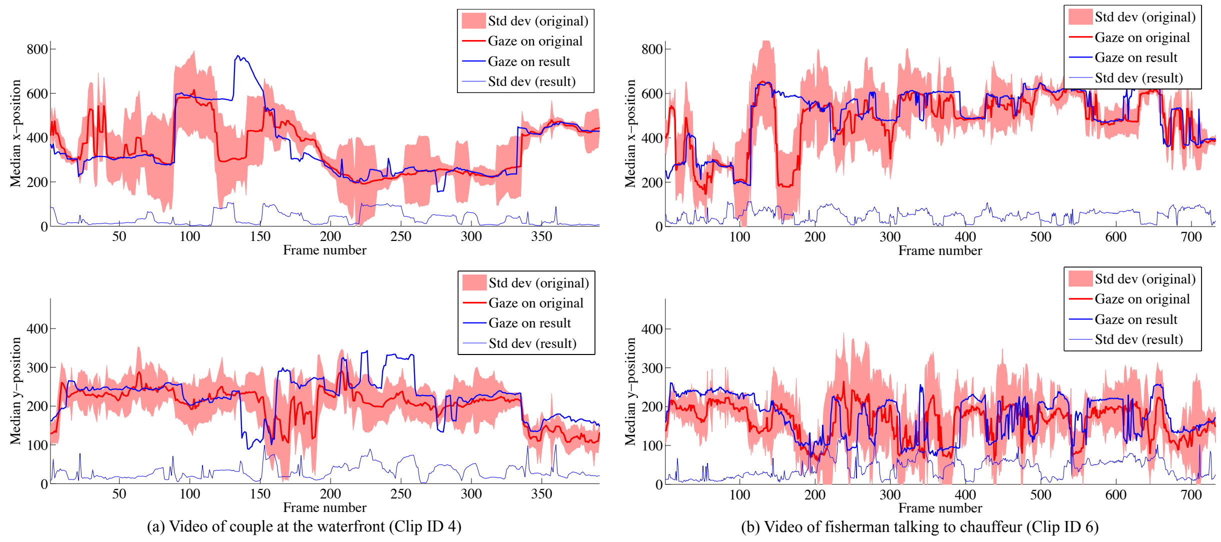


Figure 4.14: The gaze data captured on the original widescreen video is shown in red and the data captured on our result (transformed to widescreen coordinates) is shown in blue.

Chapter 5

Conclusion

The forms of visual storytelling (still pictures, animations, films) are complex and varied, but they have one common characteristic: they are made to be viewed and enjoyed by someone. Common wisdom among storytellers is that knowing their audience will help them tell their stories in a more compelling way. Computer graphics algorithms create visual stories too, either automatically, or by assisting artists. Why is it so hard to create algorithms that perform like artists do? Because artists know their audience, and computers do not.

Artists are also human, so they can process the semantics of pixels in the same way as viewers, recognizing for example, that this set of pixels represents a walk. They are trained to think about what the viewer's reaction will be, for example, that the viewer will think the character is happy if it walks in this manner. Finally, artists learn to direct the viewer's reaction, for example, they might put a spring in the step to draw the viewer's attention to the happiness of the character.

A computer does not have the ability to replicate these processes because researchers are still in the early stages of developing a computational understanding of the viewer and the artist. In this thesis, we have explored how computer graphics algorithms can 'know' their audience without waiting for computational modules to become sophisticated enough to replicate human cognition. In the first part of the work, we showed that introspecting about viewers' attentional priorities helps in the selection of good optimization terms. In the second part of the work, we presented algorithms that are driven directly with measurements of the viewers' attention.

All the presented algorithms aim to meld the traditional form of the art with the computer generated form. By 'traditional' we mean artwork that was created before computers became mainstream (such as printed comics), or artwork that can be created in the way it was done before computers (such as hand-drawn animation and live action films). Computers give us the ability to create new effects in traditional art; all we need are algorithms that can bridge the gap between the traditional and the computer-generated media. This thesis presents a collection of algorithms that bridge the gap by leveraging what the viewer cares about.

Summary

In Chapter 2, we demonstrated the design of viewer guided optimizations terms for a novel application: connecting hand-drawn and computer-generated (CG) animation to take advantage of each of their unique strengths, despite the different physical layers of each animation medium. As shown in Figure 5.1, traditional 2D animation is created by manipulating hand-drawn lines and shapes, while computer-generated 3D animation is created from 3D virtual geometric models. We connected these two media via level-of-detail dependent 3D proxies, a result of the insight that viewers will care about different attributes at the different levels of detail. We showed that physical simulation (a strength of CG animation) can be used to create secondary effects for a hand-drawn character, and pencil-on-paper rough sketches (a strength of hand animation) can be used to drive a CG skeleton.

In Chapter 3, we examined where the viewer allocates his or her visual attention while looking at a still picture and found that the eye movements of viewers can serve as a semantic driving signal for a computer algorithm. We showed that viewer gaze data can be used to predict the parameters of a three degree-of-freedom camera move across a still picture. This algorithm allowed a set of comic panels to be converted into a moves-on-stills video, a hybrid of comics and movies called 'motion comics'. In Chapter 4, we presented an algorithm to re-edit a video of a given

Medium	2D animation	3D animation	Shadow puppetry	Stop motion animation	Animatronics
Look-and-feel	flat, fluid	textured, solid	flat	organic	robotic
Physical layer/ what-is-manipulated	hand-drawn lines and shapes	3D virtual geometric models	physical 2D shapes	physical armature	electro-mechanical controls
Example	Snow White and the Seven Dwarfs	Toy Story	The Adventures of Prince Achmed	Coraline	Jurassic Park III

Figure 5.1: Differences between some popular animation media.

aspect ratio to fit a different (smaller) aspect ratio, using viewer eyetracking data as input. We also examined whether it is possible to derive a quantitative measure for the fidelity of a video manipulation from the eye movements of viewers before and after this manipulation; a subjective decision of 'this is good enough' would become a behavioral or action-based decision.

Future Directions

An interesting direction of future research is to enable more interactions between the various components of hand-drawn and computer-generated animations. Squash and stretch is a technique used by hand animators to emphasize chosen aspects of a performance. The various types of squash and stretch include shortening or extending limbs and squeezing or bulging soft tissue well beyond physically valid limits. Transferring such an effect from hand drawings to 3D characters would require an attention guided algorithm because it is not only a physics-driven effect. On the other hand, lighting is a physics-based effect, and lighting programs are used to create beautifully detailed renders for CG scenes at great expense. It is well known that humans are not very sensitive to the accuracy of shading and cast shadows. We could create similar effects for hand-drawings by generating attention guided 3D proxies. Understanding viewer attention could also improve efficiency by adaptively adjusting the resolution of lighting computations.

Just as viewer attention is a handle for connecting these two media, it could be applied towards connecting shadow puppetry, stop motion animation and animatronic animation too. Figure 5.1 shows the different forms with their physical layers, i.e., the 'stuff' that is manipulated by the artist to create a character's performance. Stop motion animation is a medium that offers an organic look and feel to the character, but generating secondary effects is cumbersome. Connecting it with CG animation might allow animators to use simulated splashes in stop motion animation instead of substituting cellophane paper for water. Animatronic animations often lack the fluidity of hand-drawn animation. Incorporating viewer attention into an algorithm might allow the identification of those portions of the animation that must be modified when transferring a hand-drawn character's performance to an animatronic body. Applications of such algorithms range from film, television and theme parks to interactive museums and video conferencing with social avatars.

For viewer attention to be a viable input variable for computer graphics algorithms, we need

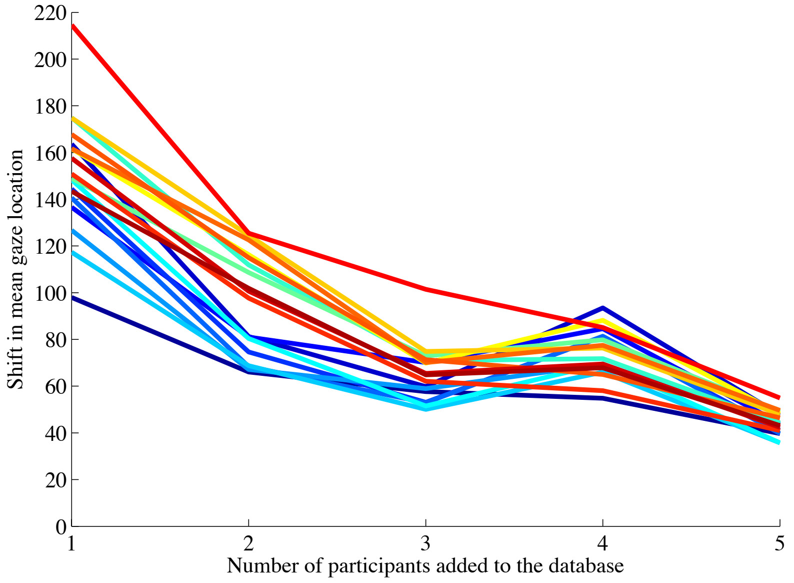


Figure 5.2: The shift in mean gaze location levels off at 4-5 viewers. The different colors are the different videos clips.

to estimate how many viewers should be eyetracked to access the “canonical” eye movement pattern. The allocation of visual attention is partially determined by bottom up cues, which are the characteristics of the stimuli that make certain portions of it more conspicuous than others, and partially by top-down mechanisms, which are specific to individual viewers. That does not mean that every individual will parse a visual stimulus (whether image or video) in an entirely new way: because our semantics are shared. As shown in Chapter 3, the eye movements of viewers are significantly more consistent on artist-created images (comic books) compared to non-artist-created images (taken by a robot for example). We computed the change in mean gaze location per frame of a video clip as the number of viewers in the database was increased. As illustrated in Figure 5.2, the shift in mean gaze location levels off at around four or five viewers. The numeric value will not reach zero because it is not necessary that the true distribution of gaze locations is drawn from a single Gaussian distribution. It could be a mixture distribution, if there were two people talking, for example.

A related question is whether children and adults follow the same viewing patterns, because it is not obvious that children have the same shared semantics as adults. We might find larger inter-observer variability among child participants compared to adult participants. Perhaps experiments will show that low level salience like sharp contrast has a greater influence on children, or perhaps that children attend to the face of the talking character more than the listening character or other objects that are discussed in the conversation. If this were the case, then instead of eyetracking adults to drive computer-generated augmentations to children’s media, we would need to recruit child participants.

The eyetracking device thus becomes a probe by which an algorithm may access what is important to the viewer, without requiring the viewer to introspect or be specially trained. Further, eyetracking in a natural environment allows us to gather data without the tedium of mouse-click-based annotation. With the availability of smaller and lighter eyetracking devices, we could potentially have algorithms that crowdsource judgements to viewers. For example, algorithms could tune internal parameters by collecting viewer gaze data, either iteratively for each parameter, or perhaps even in parallel if sufficient viewers are available.

As eyetracking technologies become less intrusive, every time a viewer watches a stimuli, corresponding gaze data can be collected. Video and images could be stored with this data as a new channel (RGB-V), allowing algorithms to access not only the information contained in the pixels of the video or image, but also information about how viewers parsed these pixels. Then, for example, a ranking algorithm could rate given films by processing the viewing patterns in addition to critics' scores.

This scale of data collection might allow us to have access to enough data under natural viewing conditions to better understand how viewing patterns change with multiple viewings, with age (children versus adults), and with changing times (the 'MTV-effect' or the 'Twitter-age attention span') for example. Real-time availability of this data could also allow algorithms to make viewer-driven selections of which camera feed to broadcast in talk shows, sporting events and concerts. Other applications include creating more 'LabelMe'-type datasets driven entirely from viewer gaze data, without the tedium of mouse-clicking.

This data could also be used to adjust amateur videos such as home videos or youtube videos to better follow cinematic conventions. A massive database of viewer eyetracking data for a variety of scenes (both in 2D, such as paintings or comic book images, and in 3D, such as Maya scenes) could allow algorithms to learn how the arrangement of objects affects the flow of viewer attention. Further, algorithms could learn arrangements that are 'good' for artistic compositions versus technical illustrations versus data visualizations. Thus, we could build tools that guide amateur artists during the creation of pictures or scenes. This direction also raises human-computer interaction questions: what is the right language for such a tool to communicate with the amateur during the creation process?

Automatic algorithms to augment or otherwise manipulate visual data need an evaluation criterion to measure the quality of the result. We have presented an evaluation criterion that compares eye movements on the original video and the retargeted result to determine the quality of the video retargeting operation. While it is always undesirable to manipulate video from the form in which in was created by the artist, our evaluation criterion provides a quantitative metric to measure the quality of the resulting video: the best possible representation of the original video after the chosen manipulation should be the one that best preserves viewer attention.

This criterion provides a necessary condition for any video manipulation to be effective. If viewers did not foveate at the same regions in each frame of the video clip after the video has been processed, the new video is definitely different from the original video. The difference could be because the regions that were attended to in the original video are no longer included in the result, or because visual artifacts in the result video drew the viewers' attention away from the originally attended regions. Note that the criterion is necessary but not sufficient to establish that the resulting video is the best possible representation of the original video. For example, if the region of a face that was attended in the original video is shown in the resulting video but the entire face is not, or if the face is shown but is made tall and skinny, the eye movements may be unchanged but the resulting video may still be considered suboptimal because distorting or cropping faces is aesthetically undesirable.

Interestingly, it might be appropriate to think of viewer gaze as the visual analogue to motion capture data. Human motion is the product of a complex neuromuscular control system. Motion capture technology has allowed us to record the output of this control system. Computer graphics researchers have been able to mine the recorded data not only to learn the parameters underlying the various control processes, but also as a driving signal (for performance animation and creating novel motions from a pre-recorded database, for example), as a regularizer (for physical simulations based on simple models, for example), and as a gold standard (for evaluating the quality of algorithmically generated human motion). Through this work, we suggest that gaze data can similarly serve as a driving signal, regularizer and metric for evaluation.

Though it is a rich source of information, gaze data only records where the viewer is foveating, which is a subset of where the viewer is attending. For attention guided algorithms to achieve the same level of sophistication as a human artist, they would need to be able to process not just foveation but all components of attention. An immediate example of the portion of the viewing experience not captured by measured foveation is negative space. Negative space is the space around and between the main objects of interest in a scene and is believed to contribute as much to the overall viewing experience as the objects themselves. As an example, the viewer may foveate on the faces of the persons engaged in a dialogue, but may have different reactions to the scene if the faces were separated by half the screen's width or if their noses almost touched each other. Or, if divers were photographed leaping into the edge of the frame, compared with leaping into the wide open area in front of them, the viewer might feel uneasy versus uplifted. Our work addresses the information contained in the positive space of the image and assumes that negative space is expendable. A necessary direction of future research in attention guided algorithms would be to consider the impact of negative space and examine how it may be measured and preserved during digital manipulations to artist-created content.

Appendix A

The 3D position of marker j in frame i is expressed in homogenous coordinates, $\mathbf{X}^w_{ij} = [X^w_{ij}, Y^w_{ij}, Z^w_{ij}, 1]^T$. Its projection on the image plane via the projection operator $\mathbf{M}_i$ is defined up to scale,

$$\mathbf{x}^{proj}_{ij} \cong \mathbf{M}_i \tilde{\mathbf{X}}^w_{ij}.$$

The input-match term e_a is defined as

$$e_a = ||\tilde{\mathbf{x}}_{ij} - \mathbf{x}^{proj}_{ij}||. \tag{A.1}$$

This error term can be linearized though a cross product [42]. Equation A.1 is equivalent to

$$\tilde{\mathbf{x}}_{ij} \times \mathbf{M}_i \mathbf{X}^w_{ij} = 0. \tag{A.2}$$

On rearranging the cross product as a matrix operation,

$$\mathbf{CM}_i \begin{bmatrix} X^w_{ij} \\ Y^w_{ij} \\ Z^w_{ij} \\ 1 \end{bmatrix} = 0, \tag{A.3}$$

where $\mathbf{C} = \begin{bmatrix} 0 & -1 & \tilde{y}_{ij} \\ 1 & 0 & -\tilde{x}_{ij} \\ -\tilde{y}_{ij} & \tilde{x}_{ij} & 0 \end{bmatrix}$, and $\mathbf{M} = \begin{bmatrix} m_1^T \\ m_2^T \\ m_3^T \end{bmatrix}$, are known matrices. Intuitively, Equation A.3 constrains $\mathbf{X}^w_{ij}$ to lie on a back-projected ray starting at the camera center, going through the image plane point $\tilde{\mathbf{x}}_{ij}$.

We stack Equations A.3, 2.5 and 2.7 for each frame, and denote the combined linear system $\mathbf{A}_{ij}$,

$$\mathbf{A}_{ij}\mathbf{X}^w_{ij} = \mathbf{b}_{ij}. \tag{A.4}$$

The smoothness term in Equation 2.6 is incorporated into a large sparse matrix as follows:

$$\mathbf{W} \begin{bmatrix} \begin{bmatrix} \mathbf{A}_{11} & 0 & ... & ... \\ 0 & \mathbf{A}_{21} & ... & ... \\ . & . & ... & ... \\ . & . & .... & \mathbf{A}_{KN} \end{bmatrix} \\ \begin{bmatrix} \mathbf{I} & -\mathbf{I} & 0 & ... \\ .. & \mathbf{I} & -\mathbf{I} & ... \\ 0 & ... & ... & ... \\ 0 & ... & \mathbf{I} & -\mathbf{I} \end{bmatrix} \end{bmatrix} \begin{bmatrix} \mathbf{X}^w_{11} \\ \mathbf{X}^w_{21} \\ ... \\ \mathbf{X}^w_{KN} \end{bmatrix} = \begin{bmatrix} \mathbf{b}_{11} \\ \mathbf{b}_{21} \\ ... \\ \mathbf{b}_{KN} \\ \mathbf{0} \\ ... \\ \mathbf{0} \end{bmatrix}, \tag{A.5}$$

$$\mathbf{W}\mathbf{A}_{full}\mathbf{X}^{w}_{full} = \mathbf{b}_{full}, \tag{A.6}$$

where $\mathbf{W}$ is the weight matrix that describes the relative weights between the geometric constraints and the smoothing terms. We solve for the least squares solution to Equation A.6.

Bibliography

[1] William Abbot and Faisal Aldo. Ultra-low cost eyetracking as an high-information throughput alternative to BMIs. *BMC Neuroscience*, 12, 2011. 4.2, 4.6

[2] Ankur Agarwal and Bill Triggs. Recovering 3d human pose from monocular images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(1):44–58, 2006. 2.2.2

[3] Javier San Agustin, Henrik Skovsgaard, Emilie Mollenbach, Maria Barret, Martin Tall, Dan Witzner Hansen, and John Paulin Hansen. Evaluation of a low-cost open-source gaze tracker. In *ETRA*, pages 77—80, 2010. 4.2, 4.6

[4] James R. Antes. The time course of picture viewing. *Journal of Experimental Psychology*, 103(1), 1974. 3.7

[5] Kohei Arai and Herman Tolle. Automatic e-comic content adaptation. *International Journal of Ubiquitous Computing*, 1, 2010. 3.2

[6] Shai Avidan and Ariel Shamir. Seam carving for content-aware image resizing. *ACM Transactions on Graphics*, 26(3), July 2007. 4.2

[7] Ilya Baran, Daniel Vlasic, Eitan Grinspun, and Jovan Popović. Semantic deformation transfer. *ACM Transactions on Graphics*, 28(3):1–6, 2009. 2.6

[8] William Bares. Panel beat: Layout and timing of comic panels. In *Smart Graphics*, volume 5166 of *Lecture Notes in Computer Science*, pages 273–276. 2008. 3.2

[9] Lubomir Bourdev and Jitendra Malik. Poselets: Body part detectors trained using 3d human pose annotations. *IEEE International Conference on Computer Vision*, 2009. 2.2.2

[10] Rebecca Boyle. DIY EyeSeeCam tracks your eyes' movement to film what you see. Popsci, May 26 2010. 1

[11] Rebecca Boyle. EyePhone: New cellphone software tracks users' eye movements for control. Popsci, May 24 2010. 1

[12] C. Bregler and J. Malik. Tracking people with twists and exponential maps. *IEEE Conference on Computer Vision and Pattern Recognition*, 1998. 2.2.2

[13] Ran Carmi. *Attention, Movie Cuts, and Natural Vision: A Functional Perspective*. PhD thesis, University of Southern California, 2007. 3.2

[14] Susana Castillo, Tilke Judd, and Diego Gutierrez. Using eye-tracking to assess different image retargeting methods. In *Symposium on Applied Perception in Graphics and Visualization (APGV)*, 2011. 4.1, 4.2

[15] Chung Chan, Howard Leung, and Taku Komura. Automatic panel extraction of color comic images. In *Advances in Multimedia Information Processing PCM 2007*, volume 4810 of *Lecture Notes in Computer Science*, pages 775–784. 2007. 3.2

[16] Li-Qun Chen, Xing Xie, Xin Fan, Wei-Ying Ma, Hong-Jiang Zhang, and He-Qin Zhou. A visual attention model for adapting images on small displays. *Multimedia Systems*, 2003. 3.2, 4.2

[17] Forrester Cole, Aleksey Golovinskiy, Alex Limpaecher, Heather Stoddart Barros, Adam Finkelstein, Thomas Funkhouser, and Szymon Rusinkiewicz. Where do people draw lines? *Communications of the ACM*, pages 107–115, Jan 2012. 1

[18] Cristina Conati, Christina Merten, Saleema Amershi, and Kasia Muldner. Using eye-tracking data for high-level user modeling in adaptive interfaces. In *Proceedings of the 22nd national conference on Artificial intelligence - Volume 2*, pages 1614–1617, 2007. 3.2

[19] Doug Cooper. 2D/3D Hybrid character animation on "Spirit". *ACM SIGGRAPH '02 Conference Abstracts and Applications*, 2002. 2.2.1

[20] Wagner Toledo Corrêa, Robert J. Jensen, Craig E. Thayer, and Adam Finkelstein. Texture mapping for cel animation. *ACM SIGGRAPH*, pages 435–446, 1998. 2.2.1

[21] Shamus Culhane. *Animation From Script to Screen*. St. Martin's Press, New York, 1990. 2.3.1

[22] Eric Daniels. Deep canvas in Disney's Tarzan. *ACM SIGGRAPH*, page 200, 1999. 2.2.1

[23] James Davis, Maneesh Agrawala, Erika Chuang, Zoran Popovic, and David H. Salesin. A sketching interface for articulated figure animation. *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 320–328, 2003. 2.2.1

[24] Doug DeCarlo and Anthony Santella. Stylization and abstraction of photographs. *ACM Transactions on Graphics*, 21(3):769–776, 2002. 1, 3.2

[25] D. Demirdjian, T. Ko, and T. Darrel. Constraining human body tracking. *IEEE International Conference on Computer Vision*, 2003. 2.2.2

[26] Thomas Deselaers, Philippe Dreuw, and Hermann Ney. Pan, zoom, scan. *Computer Vision and Pattern Recognition*, pages 1–8, 2008. 4.2

[27] Michael Devyver, Akihiro Tsukada, and Takeo Kanade. A wearable device for first person vision. In *Quality of Life symposium, Festival of International Conferences on Caregiving, Disability, Aging and Technology*, Toronto, Canada, June 2011. 1

[28] Edward Dmytryk. *On film editing*. Focal Press, 1984. 1, 4.3.3

[29] Steven M. Drucker and David Zeltzer. Camdroid: a system for implementing intelligent camera control. In *Symposium on Interactive 3D graphics*, I3D, pages 139–144, 1995. 3.2

[30] Steven M. Drucker, Tinsley A. Galyean, and David Zeltzer. Cinema: a system for procedural camera movements. In *Symposium on Interactive 3D Graphics*, pages 67–70, 1992. 3.2

[31] Jacobs David E., Goldman Dan B., and Shechtman Eli. Cosaliency: where people look

when comparing images. In *ACM Symposium on User interface software and technology*, pages 219–228, 2010. 3.2

[32] Will Eisner. *Comics and Sequential Art*. W.W. Norton & Company, 2008. 1, 3.1

[33] A. Elgammal and C. Lee. Inferring 3d body pose from silhouettes using activity manifold learning. *IEEE Conference on Computer Vision and Pattern Recognition*, 2004. 2.2.2

[34] Dan Ellis. Dynamic time warp (DTW) in Matlab . www.ee.columbia.edu/ dpwe/resources/matlab/dtw/, 2003. 2.3.1, 3.3

[35] M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Comm. Assoc. Comp. Mach.*, 24:381—395, 1981. 4.3.2

[36] James D. Foley, Andries van Dam, Steven K. Feiner, and John F. Hughes. *Computer Graphics Principles and Practice*. Addison-Wesley Publishing Company, 2 edition, 1996. 4.3.2

[37] David A. Forsyth, Okan Arikan, Leslie Ikemoto, James O'Brien, and Deva Ramanan. Computational studies of human motion: part 1, tracking and motion synthesis. *Foundations and Trends in Computer Graphics and Vision*, 1(2-3):77–254, 2005. ISSN 1572-2740. 2.2.2

[38] Ran Gal, Olga Sorkine, and Daniel Cohen-Or. Feature-aware texturing. In *Proceedings of Eurographics Symposium on Rendering*, pages 297–303, 2006. 4.2

[39] Michael Gleicher. Retargetting motion to new characters. *ACM SIGGRAPH*, pages 33–42, 1998. 2.6

[40] Michael Gleicher and Andrew Witkin. Through-the-lens camera control. In *ACM SIGGRAPH*, pages 331–340, 1992. 3.2

[41] K. Grochow, S. L. Martin, A Hertzmann, and Z. Popovic. Implicit surface joint limits to constrain video-based motion capture. *ACM Transactions on Graphics*, 23(3):522–531, 2004. 2.2.2

[42] Richard Hartley and Andrew Zisserman. *Multiple View Geometry*. Cambridge University Press, 2 edition, 2003. A

[43] Liwei He, Michael F. Cohen, and David H. Salesin. The virtual cinematographer: A paradigm for automatic real-time camera control and directing. In *ACM SIGGRAPH*, pages 217–224, 1996. 3.2

[44] L. Herda, R. Urtasun, and P. Fua. Implicit surface joint limits to constrain video-based motion capture. *European Conference on Computer Vision*, pages 405–418, 2004. 2.2.2

[45] Anh Khoi Ngo ho, Jean-Christophe Burie, and Jean-Marc Ogier. Comic page structure analysis based on automatic panel extraction. In *Ninth International Workshop on Graphics Recognition*, 2011. 3.2

[46] Alexander Hornung, Ellen Dekkers, and Leif Kobbelt. Character animation from 2d pictures and 3d motion data. *ACM Transactions on Graphics*, 26(1):1–9, 2007. 2.3.3

[47] Sarah Howlett, John Hamill, and Carol O'Sullivan. Predicting and evaluating saliency for

simplified polygonal models. *ACM Transactions on Applied Perception*, 2:286–308, July 2005. 3.2

[48] Aulikki Hyrskykari, Päivi Majaranta, Antti Aaltonen, and Kari-Jouko Räihä. Design issues of iDICT: a gaze-assisted translation aid. In *Eye Tracking Research and Applications (ETRA)*, 2000. 3.2

[49] Leslie Ikemoto, Okan Arikan, and David Forsyth. Generalizing motion edits with gaussian processes. *ACM Transactions on Graphics*, 28(1):1–12, 2009. 2.6

[50] Eakta Jain, Yaser Sheikh, and Jessica K. Hodgins. Leveraging the talent of hand animators to create three-dimensional animation. *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2009. 2.1, 2.3.4

[51] Eakta Jain, Yaser Sheikh, Moshe Mahler, and Jessica Hodgins. Augmenting hand animation with three-dimensional secondary motion. *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2010. 2.1

[52] Halszka Jarodska and Kenneth Holmqvist. A vector-based, multidimensional scanpath similarity measure. In *Eye Tracking Research and Applications (ETRA)*, 2010. 3.3, 3.3

[53] S. Card J.D. Mackinlay and G. Robertson. Rapid controlled movement through a virtual 3d workspace. In *ACM SIGGRAPH*, pages 171–176, 1990. 3.2

[54] Ollie Johnston and Frank Thomas. *The Illusion of Life: Disney Animation*. Disney Editions; Rev Sub edition, 1995. 2.1, 2.3.1

[55] Scott F. Johnston. Lumo: Illumination for cel animation. *NPAR '02: Symposium on Non-Photorealistic Animation and Rendering*, pages 45–52, 2002. 2.2.1

[56] Tilke Judd, Krista Ehinger, Frédo Durand, and Antonio Torralba. Learning to predict where humans look. In *IEEE International Conference on Computer Vision*, 2009. 3.3, 3.3, 4.1

[57] Melih Kandemir, Veli-Matti Saarinen, and Samuel Kaski. Inferring object relevance from gaze in dynamic scenes. In *Eye Tracking Research and Applications (ETRA)*, 2010. 3.2

[58] Hongwen Kang, Alexei A. Efros, Martial Hebert, and Takeo Kanade. Image matching in large scale indoor environment. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Workshop on Egocentric Vision*, 2009. 3.3

[59] Steven D. Katz. *Film directing, shot by shot, visualizing from concept to screen*. Michael Wiese Productions, 1991. 3.4.3, 4.3.2

[60] Natasha Kholgade, Iain Matthews, and Yaser Sheikh. Content retargeting using parameter-parallel facial layers. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2011. 1

[61] Philipp Krähenbühl, Manuel Lang, Alexander Hornung, and Markus Gross. A system for retargeting of streaming video. *ACM Transactions on Graphics*, 28:126:1–126:10, December 2009. 4.2, 4.4

[62] John Lasseter. Tricks to animating characters with a computer. In *ACM SIGGRAPH Course Notes*, 1994. 1, 2.1

[63] H. J. Lee and Z. Chen. Determination of 3d human body postures from a single view. *Computer Vision, Graphics, and Image Processing*, 30:148–168, 1985. 2.2.2

[64] Jehee Lee, Jinxiang Chai, Paul S. A. Reitsma, Jessica K. Hodgins, and Nancy S. Pollard. Interactive control of avatars animated with human motion data. *ACM Transactions on Graphics*, 21(3):491–500, 2002. 2.6

[65] Yin Li, Michael Gleicher, Ying-Qing Xu, and Heung-Yeung Shum. Stylizing motion with drawings. *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 309–319, 2003. 2.6

[66] Feng Liu and Michael Gleicher. Video retargeting: Automating pan and scan. In *Media '06, ACM international conference on Multimedia*, pages 241–250, 2006. 4.1, 4.2, 4.4, 4.6

[67] Ligang Liu, Renjie Chen, Lior Wolf, and Daniel Cohen-Or. Optimizing photo composition. *Computer Graphic Forum (Proceedings of Eurographics)*, 29(2):469–478, 2010. 4.6

[68] K.R. Gegenfurtner M. Dorr, T. Martinetz and E. Barth. Variability of eye movements when viewing dynamic natural scenes. *Journal of Vision*, 10, 2010. 3.2, 3.3, 4.2

[69] Eric Marchand and Nicolas Courty. Image-based virtual camera motion strategies. In *Graphics Interface*, pages 69–76, 2000. 3.2

[70] Scott McCloud. *Understanding Comics*. HarperPerennial, 1993. 3.2

[71] Rachel McDonnell, Simon Dobbyn, and Carol O'Sullivan. LOD human representations: A comparative study. In *International Workshop on Crowd Simulation (V-CROWDS'05)*, pages 101—115, 2005. 1

[72] Rachel McDonnell, Simon Dobbyn, Steven Collins, and Carol O'Sullivan. Perceptual evaluation of LOD clothing for virtual humans. In *ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, pages 117–126, 2006. 1

[73] Rachel McDonnell, Fiona Newell, and Carol O'Sullivan. Smooth movers: Perceptually guided human motion simulation. In *ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, pages 259–270, 2007. 1

[74] Rachel McDonnell, Michéal Larkin, Simon Dobbyn, Steven Collins, and Carol O'Sullivan. Clone attack! perception of crowd variety. *ACM Transactions on Graphics*, 27(3):26:1–26:8, 2008. 1

[75] Rachel McDonnell, Sophie Joerg, Jessica K. Hodgins, Fiona Newell, and Carol O'Sullivan. Evaluating the effect of motion and body shape on the perceived sex of virtual characters. *ACM Transactions on Applied Perception*, 5(4), 2009. 1

[76] Thomas B. Moeslund and Erik Granum. A survey of computer vision-based human motion capture. *Computer Vision and Image Understanding*, 81(3):231–268, 2006. 2.2.2

[77] Walter Murch. *In the Blink of an Eye*. Silman-James Press, Los Angeles, 2 edition, 2001. 1

[78] Dan Nosowitz. Lenovo demonstrates eye-tracking laptop for stare-controlled computing.

Popsci, March 1 2011. 1

[79] David Noton and Lawrence Stark. Scanpaths in eye movements during pattern perception. *Science*, 171(3968):308–3011, 1971. 3.1, 3.4

[80] Carol O'Sullivan, Sarah Howlett, Yann Morvan, Rachel McDonnell, and Keith O'Conor. Perceptually adaptive graphics. Technical report, 2004. 1

[81] Stephen E. Palmer. *Vision Science,Photons to Phenomenonlogy*. The MIT Press, Cambridge, Massachusetts 02142, 1999. 3.1, 3.4.1, 4.5

[82] Sebastian Pannasch, Jens R. Helmert, Katharina Roth, Ann-Katrin Herbold, and Henrik Walter. Visual fixation durations and saccade amplitudes: Shifting relationship in a variety of conditions. 2, 2008. 3.7

[83] Lena Petrović, Brian Fujito, Lance Williams, and Adam Finkelstein. Shadows for cel animation. *ACM SIGGRAPH*, pages 511–516, 2000. 2.2.1, 2.3.3

[84] Flip Phillips, Morgan W. Casella, and Brian M. Gaudino. What can drawing tell us about our mental representation of shape? *Journal of Vision*, 5(8), 2005. 1

[85] M. Pomplun, H. Ritter, and B. Velichkovsky. Disambiguating complex visual information: Towards communication of personal views of a scene. *Perception*, 25:931—948, 1996. 3.3

[86] Paul Rademacher. View-dependent geometry. In *ACM SIGGRAPH*, pages 439–446, 1999. 1, 2.1

[87] D. Ramanan, D. Forsyth, and A. Zisserman. Tracking people by learning their appearance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(1):65–81, 2004. 2.2.2

[88] Ganesh Ramanarayanan, James Ferwerda, Bruce Walter, and Kavita Bala. Visual equivalence: towards a new standard for image fidelity. *ACM Transactions on Graphics*, 26(3): 76:1—76:11, 2007. 1

[89] Ganesh Ramanarayanan, Kavita Bala, and James A. Ferwerda. Perception of complex aggregates. *ACM Transactions on Graphics*, 27(3):60:1–60:10, 2008. 1

[90] Ganesh Ramanarayanan, Kavita Bala, James A. Ferwerda, and Bruce Walter. Dimensionality of visual complexity in computer graphics scenes. volume 6806. SPIE, 2008. 1

[91] Keith Rayner. Eye movements in reading and information processing: 20 years of research. *Psychological Bulletin*, 124:372—422, 1998. 3.4.1

[92] P. S. A. Reitsma, J. Andrews, and N. S. Pollard. Effect of character animacy and preparatory motion on perceptual magnitude of errors in ballistic motion. *Computer Graphics Forum*, 27(2):201–210, 2008. 1

[93] Paul S. A. Reitsma and Carol O'Sullivan. Effect of scenario on perceptual sensitivity to errors in animation. *ACM Transactions on Applied Perception*, 6(3):15:1–15:16, 2009. 1

[94] Paul S. A. Reitsma and Nancy S. Pollard. Perceptual metrics for character animation: Sensitivity to errors in ballistic motion. *ACM Transactions on Graphics*, 22(3):537–542,

July 2003. 1

[95] 2002 Lucasfilm Ltd. All rights reserved. Star wars episode II: Attack of the clones: Production notes. http://www.cinema.com/articles/854/star-wars-episode-ii-attack-of-the-clones-production-notes.phtml, 2002. 1

[96] Alec Rivers, Takeo Igarashi, and Fredo Durand. 2.5d cartoon models. *ACM Transactions on Graphics*, 29:59:1–59:7, 2010. 2.2.1

[97] L Itti RJ Peters, A Iyer and C Koch. Components of bottom-up gaze allocation in natural images. *Vision Research*, 2005. 3.3

[98] B. Robertson. Mixed media. *Computer Graphics World*, pages 32–35, December 1998. 2.2.1

[99] B. Rosenhahn, T. Brox, D. Cremers, and H.-P. Seidel. Online smoothing for markerless motion capture. *Pattern recognition – Proc. DAGM*, 4713:163–172, 2007. 2.2.2

[100] B. Rosenhahn, T. Brox, and H.-P. Seidel. Scaled motion dynamics for markerless motion capture. *IEEE Conference on Computer Vision and Pattern Recognition*, 2007. 2.2.2

[101] B. Rosenhahn, T. Brox, D. Cremers, and H.-P. Seidel. Staying well grounded in markerless motion capture. *Pattern recognition – Proc. DAGM*, 5096:385–395, 2008. 2.2.2

[102] Ruth Rosenholtz. A simple saliency model predicts a number of motion popout phenomena. *Vision Research*, 39, 1999. 4.1

[103] Michael Rubinstein, Ariel Shamir, and Shai Avidan. Improved seam carving for video retargeting. *ACM Transactions on Graphics*, 27(3):16:1–16:9, August 2008. 4.2

[104] Alla Safonova, Jessica K. Hodgins, and Nancy S. Pollard. Synthesizing physically realistic human motion in low-dimensional, behavior-specific spaces. *ACM Transactions on Graphics*, 23(3), 2004. 2.3.4

[105] Hiroaki Sakoe and Seibi Chiba. Dynamic programming algorithm optimization for spoken word recognition. *Readings in speech recognition*, pages 159–165, 1990. 2.3.1, 3.3

[106] Anthony Santella, Maneesh Agrawala, Doug DeCarlo, David Salesin, and Michael Cohen. Gaze-based interaction for semi-automatic photo cropping. In *ACM SIGCHI Conference on Human Factors in Computing Systems*, CHI '06, pages 771–780, 2006. 1, 3.2, 4.2

[107] Ariel Shamir and Shai Avidan. Seam carving for media retargeting. *Communications of the ACM*, 52(1):77–85, 2009. 4.1

[108] Ariel Shamir and Olga Sorkine. Visual media retargeting. In *ACM SIGGRAPH ASIA 2009 Courses*, SIGGRAPH ASIA '09, pages 11:1–11:13, 2009. 4.2

[109] H. Sidenbladh, M. Black, and L. Sigal. Implicit probabilistic models of human motion for synthesis and tracking. *European Conference on Computer Vision*, 2002. 2.2.2

[110] Hedvig Sidenbladh, Michael J. Black, and David J. Fleet. Stochastic tracking of 3d human figures using 2d image motion. *European Conference on Computer Vision*, pages 702–718, 2000. 2.2.2

[111] C. Sminchisescu and B. Triggs. Estimating articulated human motion with covariance scaled sampling. *IEEE Conference on Computer Vision and Pattern Recognition*, 2003.

2.2.2

[112] C. Sminchisescu, A. Kanaujia, and D. Metaxas. Discriminative density propagation for 3d human motion estimation. *IEEE Conference on Computer Vision and Pattern Recognition*, 2005. 2.2.2

[113] Jos Stam. Nucleus: Towards a unified dynamics solver for computer graphics. *IEEE International Conference on Computer-Aided Design and Computer Graphics*, pages 1—11, 2009. 2.3.5

[114] India Starker and Richard A. Bolt. A gaze-responsive self-disclosing display. In *ACM SIGCHI Conference on Human Factors in Computing Systems*, CHI '90, pages 3–10, 1990. 3.2

[115] D. Sýkora, D. Sedláček, S. Jinchao, J. Dingliana, and S. Collins. Adding depth to cartoons using sparse depth (in)equalities. *Computer Graphics Forum*, 29(2):615–623, 2010. 2.2.1

[116] T. Ishii K. Kurata T. Omori, T. Igaki and N. Masuda. Eye catchers in comics: Controlling eye movements in reading pictorial and textual media. Technical report, Keio University, 2004. 3.2

[117] B. W. Tatler. The central fixation bias in scene viewing: selecting an optimal viewing position independently of motor biases and image feature distributions. *Journal of Vision*, 7, 2007. 3.4.2

[118] Benjamin W. Tatler, Roland J. Baddeley, and Iain D. Gilchrist. Visual correlates of fixation selection: effects of scale and time. *Vision Research*, 45:643—659, 2005. 3.3, 3.3

[119] C. J. Taylor. Reconstruction of articulated objects from point correspondences in a single uncalibrated image. *Computer Vision and Image Understanding*, 80:349–363, 2000. 2.2.2

[120] Pieter J. A. Unema, Sebastian Pannasch, Markus Joos, and Boris M. Velichkovsky. Time course of information processing during scene perception: The relationship between saccade amplitude and fixation duration. *Visual Cognition*, 12, 2005. 3.7

[121] Raquel Urtasun, David J. Fleet, and Pascal Fua. Temporal motion models for monocular and multiview 3d human body tracking. *Computer Vision and Image Understanding*, 104 (2):157–177, 2006. 2.2.2

[122] Yu-Shuen Wang, Chiew-Lan Tai, Olga Sorkine, and Tong-Yee Lee. Optimized scale-and-stretch for image resizing. *ACM Transactions on Graphics*, 27:118:1–118:8, December 2008. 4.2

[123] Yu-Shuen Wang, Hongbo Fu, Olga Sorkine, Tong-Yee Lee, and Hans-Peter Seidel. Motion-aware temporal coherence for video resizing. In *ACM SIGGRAPH Asia*, pages 127:1–127:10, 2009. 4.2

[124] Yu-Shuen Wang, Hui-Chih Lin, Olga Sorkine, and Tong-Yee Lee. Motion-based video retargeting with optimized crop-and-warp. *ACM Transaction on Graphics*, 29:90:1–90:9, July 2010. 4.2

[125] Yu-Shuen Wang, Jen-Hung Hsiao, Olga Sorkine, and Tong-Yee Lee. Scalable and coherent video resizing with per-frame optimization. *ACM Transactions on Graphics*, 30(4): 88:1–88:8, August 2011. 4.2, 4.4

[126] Xiaolin Wei and Jinxiang Chai. Videomocap: modeling physically realistic human motion from monocular video sequences. *ACM Transactions on Graphics*, 29(4):1–10, 2010. 2.2.1

[127] Daniel N. Wood, Adam Finkelstein, John F. Hughes, Craig E. Thayer, and David H. Salesin. Multiperspective panoramas for cel animation. *ACM SIGGRAPH*, pages 243–250, 1997. 2.2.1

[128] David S. Wooding. Eye movements of large populations: Ii. deriving regions of interest, coverage, and similarity using fixation maps. *Behaviour Research Methods*, 2002. 3.3

[129] Y. Wu, G. Hua, and T. Yu. Tracking articulated body by dynamic markov network. *IEEE International Conference on Computer Vision*, 2003. 2.2.2

[130] Katsu Yamane, Yuka Ariki, and Jessica Hodgins. Animating non-humanoid characters with human motion data. *ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, 2010. 1

[131] Liming Zhao and Alla Safonova. Achieving good connectivity in motion graphs. *Graphical Models*, 71(4):139–152, 2009. 2.6

[132] Ke Colin Zheng, Alex Colburn, Aseem Agarwala, Maneesh Agrawala, David Salesin, Brian Curless, and Michael F. Cohen. Parallax photography: creating 3d cinematic effects from stills. In *Graphics Interface*, pages 111–118, 2009. 3.2