Exploring Photobios

Ira Kemelmacher-Shlizerman[1] Eli Shechtman[2] Rahul Garg[1,3] Steven M. Seitz[1,3]

[1]University of Washington* [2]Adobe Systems† [3]Google Inc.

Source Automatically generated transition Target

Abstract

We present an approach for generating face animations from large image collections of the same person. Such collections, which we call *photobios*, sample the appearance of a person over changes in pose, facial expression, hairstyle, age, and other variations. By optimizing the order in which images are displayed and cross-dissolving between them, we control the motion through face space and create compelling animations (e.g., render a smooth transition from frowning to smiling). Used in this context, the *cross dissolve* produces a very strong motion effect; a key contribution of the paper is to explain this effect and analyze its operating range. The approach operates by creating a graph with faces as nodes, and similarities as edges, and solving for walks and shortest paths on this graph. The processing pipeline involves face detection, locating fiducials (eyes/nose/mouth), solving for pose, warping to frontal views, and image comparison based on Local Binary Patterns. We demonstrate results on a variety of datasets including time-lapse photography, personal photo collections, and images of celebrities downloaded from the Internet. Our approach is the basis for the Face Movies feature in Google's Picasa.

CR Categories: I.3.7 [Computer Graphics]—;

Keywords: Face animation, photo collections, cross dissolve, Picasa

Links: DL PDF WEB VIDEO

1 Introduction

People are photographed thousands of times over their lifetimes. Taken together, the photos of each person form his or her visual record. Such a visual record, which we call a *photobio*, samples the appearance space of that individual over time, capturing variations in expression, pose, hairstyle, and so forth. While acquiring photobios used to be a tedious process, the advent of photo sharing tools like Facebook coupled with face recognition technology and image search are making it easier to amass huge numbers of photos of friends, family, and celebrities. As this trend increases, we will have access to increasingly complete photobios. The large volume of such collections, however, makes them very difficult to manage, and better tools are needed for browsing, exploring, and rendering them.

If we could capture *every* expression that a person makes, from every pose and viewing/lighting condition, and at every point in their life, we could describe the *complete* appearance space of that individual. Given such a representation, we could render any view of that person on demand, in a similar manner to how a light-field [Levoy and Hanrahan 1996] enables visualizing a static scene. However, key challenges are 1) the face appearance space is extremely high dimensional, 2) we generally have access to only a sparse sampling of this space, and 3) the mapping of each image to pose, expression, and other parameters is not generally known a priori. In this paper, we take a step towards addressing these problems to create interactive, animated viewing experiences from a person's photobio. We focus on the specific problem of *view interpolation*, i.e., rendering a seamless transition between two photos. As such, we naturally generalize the view-interpolation capabilities of classic image-based rendering (IBR) methods, e.g., [Chen and Williams 1993; Seitz and Dyer 1996; Levoy and Hanrahan 1996] to handle changes in expression, age, and other transformations. But while IBR methods traditionally focus on synthesizing novel images, we instead create transitions from images already in the database, and instead seek to select the right set of in-betweens. This approach is reminiscent of Snavely et al.'s work on finding paths through Internet photo collections [Snavely et al. 2008], but applied to faces instead of places. We note that faces present unique challenges because there is not a clear underlying parameterization of the photo space, unlike the case of [Snavely et al. 2008] where it was possible to construct a function mapping pose to image.

A key insight in our work is that *cross dissolving* well-aligned images produces a very strong motion sensation. While the cross-dissolve (also known as cross-fade, or linear intensity blend) is prevalent in morphing and image-based-rendering techniques, it is usually used in tandem with a geometric warp, the latter requiring accurate pixel correspondence (i.e., optical flow) between the source images. Surprisingly, the cross dissolve *by itself* (without correspondence/flow estimation) can produce a very strong sensation of movement, particularly when the input images are well aligned. We explain this effect and prove some remarkable prop-

*e-mails: {kemelmi, rahul, seitz}@cs.washington.edu

†e-mail: elishe@adobe.com

erties of the cross dissolve. In particular, given two images of a scene with small motion between them, a cross dissolve produces a sequence in which the edges move smoothly, with *nonlinear ease-in, ease-out dynamics*. Furthermore, the cross dissolve can also synthesize physical illumination changes, in which the light source direction moves during the transition. We analyze these effects and their operating ranges.

Our photobios approach takes as input an unorganized collection of photos of a person, and produces animations of the person moving continuously. The method operates best when the photo collection is very large (several hundred or thousand photos), but produces reasonable results for smaller collections. As a special case of interest, we first show results on time-lapse sequences, where the same person is photographed every day or week over a period of years. We then apply the technique to more standard image collections of the same person taken over many years, and also to images of celebrities downloaded from the Internet.

Our approach is based on representing the set of images in a photobio as nodes in a graph, solving for optimal paths, and rendering a stabilized transition from the resulting image sequence. The key issues therefore are 1) defining the edge weights in the graph, and 2) creating a compelling, stabilized output sequence. Our approach leverages automatic face detection, pose estimation, and robust image comparison techniques to both define the graph and create the final transitions. The pipeline is almost entirely automatic—the only step that requires manual assistance is to identify the subject of interest when an image contains multiple people. Automating this step is also likely possible using face recognition techniques [Berg et al. 2004].

This graph-based formulation draws on prior work for creating character animation from motion capture libraries [Kovar et al. 2002; Arikan and Forsyth 2002]. Closely related is the *spacetime faces* work of Zhang et al. [Zhang et al. 2004] who created face animations from a database of 3D face sequences captured in the lab. Their keyframe animation approach provides the same kind of functionality that we seek, and was also based on finding shortest paths on a motion graph. Also related is Goldman et al.'s video navigation approach [Goldman et al. 2008] which provides a direct manipulation interface for controlling the pose of a person's face captured from video, Bregler et al.'s approach for re-ordering frames in a video [Bregler et al. 1997] and Pighin et al.'s approach for image based animation of facial expressions [Pighin et al. 1998] by manually specifying a correspondence between features in several photos of the person and features on the 3D head model. All of these techniques depend critically on mocap or video data as input, both 1) to enable solving the frame-to-frame correspondence problem, and 2) to generate smooth transitions. Achieving similar capabilities for unstructured photo collections is a much more difficult problem due to the irregular and sparse sampling and lack of motion correspondence information.

An alternative approach is to attempt to model and parameterize each face and its variability in 3D and over time, as done by Blanz et al. [Blanz and Vetter 1999] with impressive results. However, the challenge of such a model-based approach is that it must span the full range of human shape, expression, and appearance, for all ages, ethnicities, and so forth—a very tall order. We instead address the more modest goal of rendering from photos already in the database, with the advantage of broader applicability and more automation.

Our ability to operate on unstructured photo collections is a direct result of the maturity of computer-vision based face analysis techniques that are now in widespread use in digital cameras, Google Streetview, Apple's Iphoto, etc. In the research community, we are inspired by Berg et al.'s pioneering work on *Faces in the News* [Berg et al. 2004], which automatically labels people in photographs by finding correspondences between captions and faces in the associated news photographs. Their application is complementary to ours, in that we require as input a set of pre-classified images of the same person (the output of their approach). Also related are Kemelmacher-Shlizerman et al.'s video puppeteering [Kemelmacher-Shlizerman et al. 2010], Kumar et al.'s face search [Kumar et al. 2008] and Bitouk et al.'s swapping [Bitouk et al. 2008] work, which operate robustly on large collections of images downloaded from the Internet.

We acknowledge a large body of previous work on photo browsing both in academia and industry. Indeed, face browsing and tagging has become a sensation on Facebook, and face-recognition is built-in to tools like Iphoto and Picasa. The HCI and computer vision communities have long recognized the need for better photo browsers, see for example, [Pentland et al. 1996; Rekimoto 1999; Bederson 2001; Graham et al. 2002; Huynh et al. 2005] and have explored a number of techniques for organizing and intuitively browsing photos by place, time, content, and other characteristics. In particular, a number of authors, e.g., [Rekimoto 1999; Graham et al. 2002] have explored historical collections akin to photobios. While our work also fits into this general field, we focus on the novel problem of rendering animated transitions from still photo collections. The Face Movies feature of Picasa's 3.8 [Picasa 2010] release provides an implementation of our method.

1.1 Overview

We first describe our preprocessing pipeline that takes an unstructured collection of images, aligns them and estimates pose in Section 2. Section 3 describes the construction of the face graph, and our path finding approach. Section 4 presents our analysis of the cross dissolve transformation. In Section 5 we describe our implementation of the algorithm in Picasa 3.8 and present additional results in Section 6.

2 Automatic alignment and pose estimation

This section describes the geometric alignment part of our method (illustrated in Fig. 1). The aim here is to estimate the location of the face and its pose in each picture of the image set, and enable warping each image to a canonical pose.

Each photo is preprocessed automatically (similarly to [Kemelmacher-Shlizerman et al. 2010]) by first running a face detector [Bourdev and Brandt 2005] followed by a fiducial points detector [Everingham et al. 2006] that finds the left and right corners of each eye, the two nostrils, the tip of the nose, and the left and right corners of the mouth. We ignore photos with low detection confidence (less than 0.5 in face detection and less than -3 in detection of the fiducial points). Throughout our experiments, despite variations in lighting, pose, scale, facial expression and identity, this combination of methods was extremely robust for near-frontal faces with displacement errors gradually increasing as the face turns to profile.

The next step is to detect pose, which is achieved by geometrically aligning each detected face region to a 3D template model of a face. We use a neutral face model from the publicly available spacetime faces [Zhang et al. 2004] dataset for the template. We estimate a linear transformation that transforms the located fiducial points to pre-labeled fiducials on the template model, and use RQ decomposition to find rotation and scale. We then estimate the yaw, pitch and roll angles from the rotation matrix. Given the estimated pose we transform the template shape to the orientation of the face in the image and warp the image to a frontal pose using point-set z-buffering

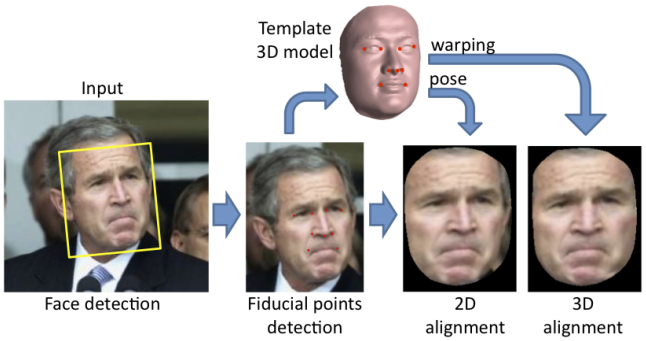


Figure 1: *Automatic alignment and pose estimation–we first apply face detector followed by a fiducial point detector. Then a 3D template model is used to estimate pose and to warp the image to a frontal view for a more consistent computation of similarities.*

[Katz et al. 2007] to account for occlusions.

3 The Face Graph

Once the images have been aligned and warped to a frontal pose, the next step is to compute a *face graph* where each face is a node and edges encode relative distances between two faces. To define face distance, we use a combination of difference in pose, appearance, and time (when timestamps are available). We first describe our distance measure, then show how to build the face graph and walk on this graph.

Distance between faces Face appearance varies drastically in between photos in general photo collections due to non rigid motion of the face, color balance, lighting changes, pose, facial hair etc. Local Binary Pattern (LBP) histograms have previously proven effective for face recognition and expression identification tasks [Ojala et al. 2002; Ahonen et al. 2006] and recently used for image based retrieval of similar facial expressions [Kemelmacher-Shlizerman et al. 2010]. LBP operate by diving an image to grid of cells and converting each pixel in a cell into a code which encodes the relative brightness patterns in a square neighborhood around that pixel. In particular, each neighbor is assigned a 1 or 0 if it is brighter or darker than the center pixel. This pattern of 1's and 0's defines a per pixel binary code, and the per cell histogram of these codes defines the descriptor for a cell. We calculate a separate set of descriptors for the eyes, mouth and hair regions, where a descriptor for a region is a concatenation of participating cells' descriptors. The regions and first four neighbors found based on comparing each of the regions separately are shown in Figure 2. The binarization quantization achieves robustness to lighting changes; robustness to saddle motions is obtained by forming the histogram. The distance between two face images i and j, denoted d_{ij}, is then defined by χ^2-distance between the corresponding descriptors, normalized using a robust logistic function $L(d) = (1 + e^{-\gamma(d-\mu)/\sigma})^{-1}$ such that $\gamma = \ln(99)$. This function normalizes the distances to the range $[0, 1]$, such that $d = \mu$ maps to 0.5 and $d = \mu \pm \sigma$ map to 0.99 and 0.01.

Our appearance distance function is then defined as:

$$D_{appear}(i,j) = 1 - (1 - \lambda^m d^m_{ij})(1 - \lambda^e d^e_{ij})(1 - \lambda^h d^h_{ij}) \quad (1)$$

where $d^{m,e,h}$ are the LBP histogram distances restricted to the mouth, eyes, and hair regions, respectively, and $\lambda^{m,e,h}$ are the corresponding weights for these regions. For example, assigning

Figure 2: *Appearance similarity is calculated separately for (a) eyes, (b) mouth and (c) hair. For each region we show the query image and its four nearest neighbors. Note how the region that is being matched looks similar.*

$\lambda^m = 1$ and $\lambda^e = \lambda^h = 0$ will result in only the mouth region being considered in the comparison. In our experiments we used $\lambda^m = 0.8$ and $\lambda^e = \lambda^h = 0.1$.

We add additional distance functions measuring L_2 difference in pose (separately for yaw and pitch), and time (when timestamps are available). Each is normalized using a robust logistic function $L(d)$, yielding distances D_{yaw}, D_{pitch}, and D_{time}.

The face graph The face graph is then defined as follows. The nodes are the faces in the dataset and edge (i, j) has weight $D(i, j)$ defined as

$$D(i,j) = \left[1 - \Pi_{s\in\{app,yaw,pitch,time\}}(1 - D_s(i,j))\right]^\alpha. \quad (2)$$

The exponent α is used to nonlinearly scale the distances, and provides additional control of step size in the path planning process.

By constructing this face graph we can now traverse paths on the graph and find smooth, continuous transitions from the still images contained in a photo collection. We do that by either look for shortest paths or by greedy walks on the face graph.

Give any two images, we can find the smoothest path between them by solving for the shortest path in the face graph. We are interested in finding a path with the minimal cost (sum of distances), which is readily solved using Dijkstra's algorithm. The number of in-between images is controlled via the α parameter.

Given any starting point, we can also produce a smooth path of arbitrary length by taking walks on the graph. Stepping to an adjacent node with minimal edge distance generally results in continuous transitions. There are a number of possible ways to avoid repetitions, e.g., by injecting randomness. We obtained good results simply by deleting previously visited nodes from the graph (and all of their incident edges). For collections with time/date information, we encourage chronological transitions by preferentially choosing steps that go forward in time.

4 The Cross Dissolve

Having produced a sequence of images, we would like to render compelling transitions from one photo to the next. Morphing techniques can produce excellent transitions, but require accurate correspondence between pixels in the images, which is difficult to obtain. A simpler alternative is to use a *cross dissolve*. The *cross dissolve*

Figure 3: *Cross dissolve synthesizes motion. Notice how the edges of the nose and mouth move realistically, as does the lighting (more clearly seen in accompanying video).*

or *cross fade* transitions between two images (or image sequences) by simultaneously fading one out while fading the other in over a short time interval. Mathematically, the cross dissolve is defined as

$$I_{out}(t) = (1-t)I_{in_1} + tI_{in_2}, \tag{3}$$

where I_{in_1} and I_{in_2} are the input images and $I_{out}(t)$ is the output sequence. This effect is often combined with geometric warps in morphing [Seitz and Dyer 1996; Beier and Neely 1992], and image-based rendering methods [Levoy and Hanrahan 1996], to synthesize motion between photos. More surprisingly, the cross dissolve *by itself* (without correspondence/flow estimation) can produce a very strong sensation of movement, particularly when the input images are well aligned. For example, Figure 3 shows a cross dissolve between two photos of a person's face, in which both the lighting and features appear to move realistically (much better seen in the accompanying video). While it makes sense that warping an image produces a motion sensation, why would motion arise from a simple intensity blend? We explain this effect and prove some remarkable properties of the cross dissolve in this section.

We show that the cross dissolve produces not just the illusion of motion, but true motion; given two images of a scene with small motion between them, a cross dissolve produces a sequence in which the edges move smoothly, with *nonlinear ease-in, ease-out dynamics*. Furthermore, the cross dissolve can also synthesize physical illumination changes, in which the light source direction moves during the transition. We now describe the mathematical basis for these effects and define their operating range.

4.1 Edge Motion

Rapid spatial changes in irradiance produce image edges. Real image edge profiles tend to be smooth rather than discontinuous, due to the optical blurring effects of the imaging process [Marr and Hildreth 1980; Nalwa and Binford 1986]. Indeed, the convolution of a step-edge with a Gaussian blurring kernel[1] is the *erf* function: $erf(x) = \int_0^x e^{-t^2}\,dt$. This function is very closely approximated as a segment of a sine curve, as shown in Figure 4. We therefore use this sine edge model for the remainder of this section.

The properties of displaced sine curves have been well-studied in the perception literature. In particular, displaying slightly displaced sine curves or other patterns in rapid succession produces a strong *apparent motion* sensation [Wertheimer 1912; Kenkel 1913; Adelson and Movshon 1982; Lu and Sperling 2002]. Freeman et al. [Freeman et al. 1991] leverage this effect to produce *motion without movement* by displaying an image followed by a filtered version.

[1] We note that the Gaussian is an imperfect PSF model [Joshi et al. 2008], but still useful as a first-order approximation.

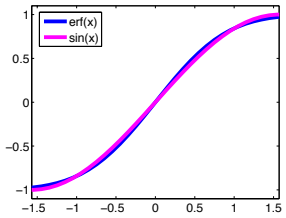


Figure 4: *A Gaussian-convolved step edge (erf) is well-approximated by a sine curve.*

While related, we note that these perception effects are fundamentally different than the cross dissolve; in the former case, our brains are doing the interpolation—in the latter, the computer does the interpolation. In the case of the cross dissolve, we can mathematically analyze and prove properties of the underlying motion (a topic not covered in the perception literature). We note that local edge models are also used in the optical flow literature to link between image intensity changes and motion [Lucas and Kanade 1981], although both the models (linear rather than sinusoid) and the applications (flow computation rather than synthesis) are quite different.

Consider two sine waves (each represents a different image) where one is a translated (and optionally amplitude-scaled) version of the other. Specifically we consider $\alpha\sin(mx)$ and $\sin(mx+d)$ so that d is the phase shift (spatial translation) and α is the amplitude scale. Cross dissolving these two sine waves produces a sequence of sine waves given as follows:

$$(1-t)\alpha\sin(mx) + t\sin(mx+d) = c\sin(mx+k) \tag{4}$$

where $t \in [0,1]$ and

$$k = \arctan\frac{t\sin d}{(1-t)\alpha + t\cos d} \tag{5}$$

$$c^2 = \alpha^2(1-t)^2 + t^2 + 2(1-t)\alpha t\cos d. \tag{6}$$

Therefore, cross dissolving two sines with different phases produces a motion, where the phase k is smoothly interpolated. This simple analysis gives rise to a number of remarkable observations:

- The speed of the motion is determined by the phase k. Note that k is not linear, but resembles the *ease-in, ease-out* curves long favored by animators [Lasseter 1987]. This type of curve is known to have a major role in producing more believable animations; it is remarkable that it arises naturally in the cross dissolve. Furthermore, different edges move at different rates, and with different ease-in/ease-out parameters, depending on their phase offsets. In particular, large displacements give rise to more exaggerated ease-in/ease-outs (see Fig. 6 (a)).
- The perceived motion is strictly less than a half-period. Hence, low-frequency edges (lower m) can move relatively large distances, whereas high frequency edges can move only slightly. When the phase offset reaches π (a half period), the edge disappears entirely at the center frame, and becomes a constant function. This phenomenon, in which image content fades away during a transition, is known as *ghosting* [Szeliski and Shum 1997].
- There is a gradual decrease in image contrast towards the midpoint of the transition, due to the drop in amplitude of the sine, according to c in Eq. (6). This is illustrated in Fig. 6 (b). For example, the highlights get darker, and the shadows get lighter. This reduction in dynamic range is subtle (except

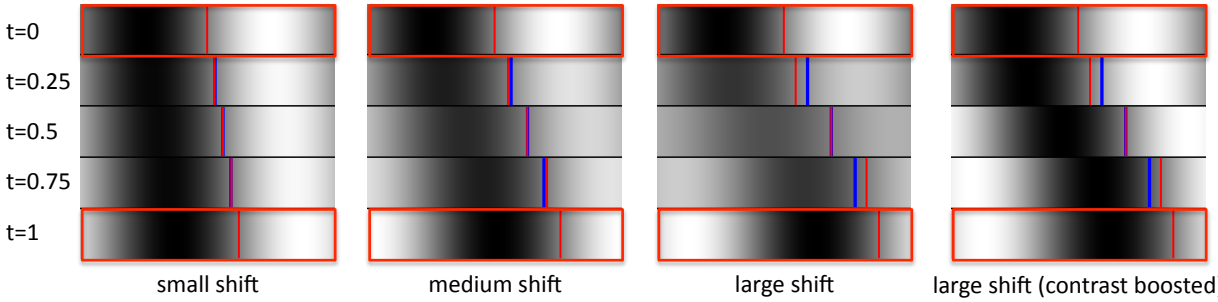
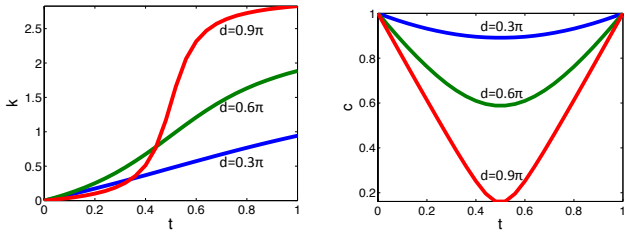


Figure 5: *Cross dissolve of sine and phased-shifted sine, with small* (0.3π)*, medium* (0.6π)*, and large* (0.9π) *shifts. We show film strips of the cross-dissolve, with rows corresponding to times* $t = 0$ *(input frame),* $.25$*,* $.5$*,* $.75$*, and* 1 *(input frame). The location of the edge is marked in red and the location corresponding to a linear motion is marked in blue. The displacement of the red and blue lines for larger shifts demonstrate the nonlinear ease-in, ease-out speed curves (better seen in the supplemental video). Also note the decrease in contrast for larger shifts. To better visualize the (nonlinear) edge motion (for a* 0.9π *shift), we remove the contrast change (far right) by scaling the image by the inverse of* c*.*



(a) Speed curve (b) Change in contrast

Figure 6: *(a) Motion speed* $k(t)$ *and (b) change in contrast* $c(t)$*, while cross dissolving* $sin(x)$ *and* $sin(x + d)$*.* t *is time (frame) of the transition and* d *defines the shift (phase offset) of the second curve relative to the first. Observe how the speed curve* $k(t)$ *resembles the ease-in ease-out curves used in traditional animation. The gradual decrease in contrast* $c(t)$ *towards the mid point visually hides artifacts.*

in the most extreme cases), yet serves to hide visual artifacts like ghosting [Szeliski and Shum 1997] in the frames in which they are most likely to appear.

- This motion effect only works for edges with (approximately) the same frequency. Interpolating sines with different frequencies produces multi-model curves that do not resemble edges (another form of ghosting).

Figure 5 illustrates these effects for sine gratings with different relative phase shifts. The contrast reduction and the speed nonlinearity are imperceptible when the shift is small, and becomes more noticeable for larger shifts. The speed nonlinearity is visible as the gap between the red and blue lines (better seen in the video).

To better visualize the speed changes, Fig. 5 (far right column) also shows an intensity-compensated transition, where each image is scaled by $1/c$ (c is given in Eq. (6), to remove the loss in contrast. These effects are also demonstrated in the supplemental video.

4.1.1 Non-periodic edges

Note that our analysis so far is based on a periodic function (sine); however, most edges are not periodic. Periodicity is not necessary, however, as the analysis applies *locally*. I.e., consider any local window of pixels; if the pixels in that window are well-approximated by a sine in both images, then all of the above conclusions apply. In particularly, consider two rising edges (modeled as *erf*: step edges convolved with a Gaussian blur kernel), where the second edge has been shifted a small amount relative to the first. As seen in Fig. 8, in the intervals where the two edges are both rising (area C), the sine approximation holds, and the cross-dissolve produces a translation. The cross-dissolve also works correctly in the intervals (A and E) in which both curves are flat. However, the intervals where one curve is rising and the other is flat (B and D) do not translate–they simply fade out. Hence, the translation effect is less pronounced for larger shifts (where the interval spanned by C is small compared to B and D). Figure 7 illustrates the effects of cross dissolving erf-edges (see video as well).

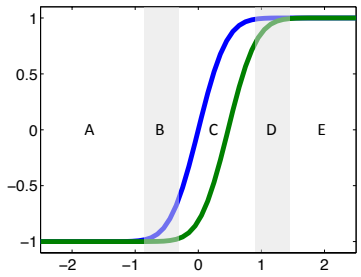


Figure 8: *The model works for the light regions (A,C,E), but ghosting appears in the gray regions (B,D).*

4.1.2 Generalizing to 2D

So far we analyzed 1D edges, however our analysis naturally generalizes to translations of 2D image edges. In case of 2D edge translation, we simply define our edge profiles in the direction normal to the edge, thus reducing to the 1D case.

We have also observed that cross dissolving two edges with different orientations also produces a compelling apparent rotation perception (see video for examples), particularly when the orientation change is small. We can explain this effect as follows. A cross dissolve of a sine and its rotated version is defined as

$$(1-t)\sin(mx) + t\sin(mx\cos\theta - my\sin\theta) \qquad (7)$$

where θ is the angle of rotation. For, two edges with angle of rotation θ between them, we can also say that one of them has been

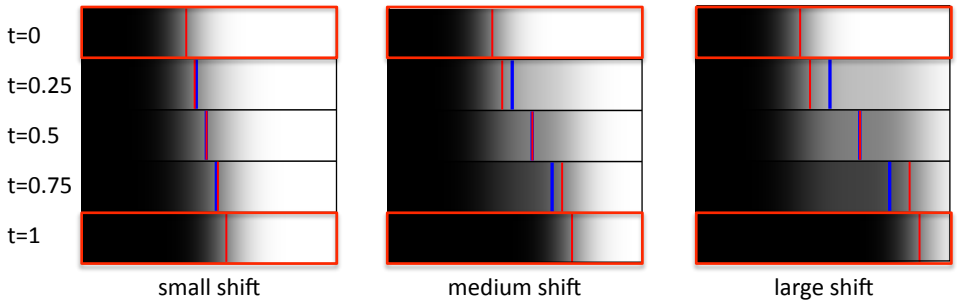


Figure 7: *Cross fade of a rising edge and its shifted version with small* (0.4π)*, medium* (0.8π) *and large* (1.2π) *shifts. An edge is approximated with the erf*$(3x)$ *function. See description of the plots in Fig. 5. The edge appears to move during the cross dissolve, but also distorts away from the center, for larger motions (see text for a explanation).*

rotated by $\frac{\theta}{2}$ and the other by $\frac{-\theta}{2}$ giving us,

$$(1-t)\sin(mx\cos\frac{\theta}{2}+my\sin\frac{\theta}{2})+t\sin(mx\cos\frac{\theta}{2}-my\sin\frac{\theta}{2}) \quad (8)$$

This is equivalent to cross dissolving two crossfaded sines with frequency $m\cos\frac{\theta}{2}$ and phase difference $2my\sin\frac{\theta}{2}$ and our analysis of translation case applies, i.e., the movement is linear and the edge appears to rotate if $2my\sin\frac{\theta}{2}$ is small. It holds for low frequency edges, small rotation angles and at relatively small distances from the center of rotation, with more distortion as you move out.

4.2 Interpolation of light sources

In addition to edge motion, cross dissolves can also produce very convincing illumination changes in which the light source direction appears to move realistically during the transition. Indeed, we now describe conditions under which a cross-dissolve produces physically-correct illumination changes.

An image of a Lambertian object, ignoring shadows, specularities, and inter-reflections, is determined by $I = \rho l^T n$ where ρ is the albedo, l is the lighting direction vector and n is the surface normal vector. Cross dissolving two such images: $(1-t)I_1 + tI_2$, has the effect of interpolating the lighting directions $\rho((1-t)l_1 + tl_2)^T n$. In particular, we can rewrite the image formation equation as $I = \rho|l|\cos\phi$ where ϕ is the angle between the surface normal at each point on the surface and lighting direction.

A cross dissolve of two images can be then formulated as

$$(1-t)\rho|l_1|\cos\phi + t\rho|l_2|\cos(\phi+d) = c_l\cos(\phi+k_l). \quad (9)$$

with d being the difference between the surface normal and the two lighting direction angles. Hence, the interpolated pixel is the sum of two shifted cosines, which is also a cosine. In this case, however, the cosine is not in the image plane, but rather defines the variation of the pixel intensity as a function of lighting. Denote by $a = (1-t)\rho|l_1|$ and $b = t\rho|l_2|$. The amplitude (contrast change) is then $c_l^2 = a^2 + b^2 + 2ab\cos d$, and the motion speed is $k_l = \arctan\frac{b\sin d}{a+b\cos d}$. Note that in that case the change in amplitude and speed depends on the surface normal and the lighting intensity $|l_1|, |l_2|$ in addition to the shift d and time t.

The amplitude change results in an effective *dimming* of the light during the transition, with minimum contrast occurring at the midpoint. We experimented with ways to compensate for this contrast change, e.g., by estimating the contrast reduction and rescaling the image intensities. Surprisingly, we found that even a perfect compensation produces visually inferior results to the uncompensated cross dissolve. The reason is that this dimming effect serves to hide artifacts due to shadows, specular highlights, and other non-Lambertian effects that are not modeled by Eq. (9), whereas the compensation only magnifies these artifacts. The cross dissolve thereby hides artifacts in the frames in which they are most likely to appear—a remarkable property!

While we are presenting the specific light trajectory of the cross dissolve (two-image) case for the first time, we emphasize that the basic result that image interpolations produce new directional illuminations of Lambertian objects is well-known in the computer vision community, going back to the work of Shashua [Shashua 1992].

5 Face Movies in Picasa

The *Face Movies* feature of Picasa's 3.8 release [Picasa 2010], provides an implementation of photobios, targeted to personal photos. This feature leverages Picasa's built-in face recognition capabilities, and enables creating a face movie of a person with minimal effort (in as little as a single click).

Deploying the approach at scale (with photo collections numbering in the tens of thousands) required a number of modifications and improvements to the basic algorithm. Moreover, the Picasa version employs a different rendering style. We briefly describe the most important modifications below.

5.1 Real-time performance

An important requirement for the feature was that it should be real-time, i.e., the face movie should start playing almost immediately when the feature is selected. We achieved this goal through a number of optimizations.

First, the mouth is highly correlated with the eyes and other regions of the face in forming expressions. Hence, we found it sufficient to match only the mouth region. More surprisingly, we found that head *orientation* is also well correlated with the appearance of the mouth region (producing mouth foreshortening and rotation), eliminating the need to compute pose explicitly. Similarly, we found that using 2D affine image alignment rather than 3D warping produces satisfactory results at lower cost. These optimizations, as well as the use of HOG (Histogram of Oriented Gradients) features [Dalal and Triggs 2005] in place of LBP, significantly reduced matching costs.

By default, face movies creates and renders a greedy walk rather than an optimized path, as the former is faster to compute. We accomplish this using a multi-threaded approach where one thread

Figure 9: *In Picasa, the images are aligned and displayed by stacking them over one another.*

computes the image graph on the fly and selects the next image to step to, while the other thread renders the transition between the previous two images. For computing the greedy sequence we first sort all the images by time, start at the oldest image and then consider the next 100 images in the sequence, chronologically. We choose the one with the closest similarity as the next image to step to. This procedure repeats until the time window overlaps the most recent photo, at which point we reverse direction, i.e., select photos going monotonically back in time. We then continue to oscillate forward and back until all images have been shown. We give preference to *starred* photos by reducing their edge weights so that they are more likely to be shown toward the beginning of the face movie.

The user can control the length of the movie by specifying the number of photos in the face movie and we find the optimal sequence of desired length via dynamic programming. To achieve reasonable performance (a delay of a few seconds, even for collections of tens of thousands of images), we employed additional optimizations, such as breaking the sequence into separately-optimized chunks of 1000 images, and sparsifying the graph by considering only 100 neighbors for each image.

5.2 Visual Appearance

The Picasa feature introduces two improvements to the rendering style. First, we found that people prefer seeing more of the photos beyond just the cropped faces, as the wider field of view provides more context. Second, instead of showing photos one at a time, we layer the aligned photos over one another as shown in Figure 9. This layering process produces interesting effects where the head from one photo is superimposed with the body from another. The user interface also provides the ability to output the movie, upload to the web, or add audio, captions, and customize appearance in several other ways.

6 Results

We experimented with datasets downloaded from the Internet, and with personal photo collections. Hundreds of face movies can also be found on YouTube, created by users of Picasa.

Most of our results are best viewed in the supplemental video; we present a few example paths in Figure 10, and compare to showing aligned images without path optimization in Figure 11.

We first show results on time-lapse photo collections, in which a single person is photographed every week/day over a period of years, and usually include large variations in facial expression, hair, etc. We demonstrate results on two such collections. The "Daily Jason" dataset (http://www.supyo.com/jason/) contains 1598 pictures taken almost every day during 5 years. Figure 10 (a) shows an optimized path—the end points (marked in red) are chosen by the user in our interface and the intermediate sequence is computed by our method. Note the smooth transitions in mouth expression and eyes. The second dataset shows photos of "Pearl" photographed weekly from birth until she was eight years old (519 pictures). In Figure 11 (a) we show a path that was computed by our method and in (b) a sequence of images that was produced by uniformly sampling each time frame. Our method was able to produce a path in which the girl ages, while producing smooth changes in facial expression. In the uniformly sampled sequence we can see highly non-coherent transitions, e.g, the face or mouth is occluded by hands, eyes close and open, large jumps in the mouth expression.

We have also experimented with personal photo collections: 1) 584 pictures over 5 years of Amit 2) 1300 pictures of Ariel over 20 years, and 3) 530 photos of George W. Bush taken from the Labeled Faces in the Wild [Huang et al. 2007] collection. In contrast to the time-lapse datasets, the pictures in these three datasets were taken in arbitrary events, locations, with various illumination, resolution, cameras etc., and are therefore more challenging. Figs. 10 (b,c) and 11 (c,d) show the results. Note how in all sequences, in addition to smooth transition in facial expression, the pose changes smoothly.

Examples of Face Movies created by people can be found on YouTube, here are links to three of our favorites: Lily's photos, To the future, The Many Faces of Miley Cyrus.

7 Conclusions

We presented a new technique for creating animations of real people through time, pose, and expression, from large unstructured photo collections. The approach leverages computer vision techniques to compare, align, and order face images to create pleasing paths, and operates completely automatically. The popular photo browsing tool Picasa has an implementation of this approach, known as "Face Movies", which has seen widespread deployment. Key to the success of this method is the use of the *cross dissolve*, which produces a strong physical motion and illumination change sensation when used to blend well-aligned images. We analyzed this effect and its operating range, and showed that, surprisingly, cross dissolves do indeed synthesize true edge motion and lighting changes under certain conditions.

Future work in this area may leverage better recognition, alignment, and correspondence algorithms to yield even better transitions and with smaller photo collections (the current approach works best with image collections numbering in the hundreds or thousands). It would also be interesting to explore other ways to navigate personal photo collections, considering both individuals and groups of people that are photographed together.

Acknowledgements

This work was supported in part by National Science Foundation grant IIS-0811878, the University of Washington Animation Research Labs, Adobe, Google, and Microsoft. We thank the following people for the use of their photos: Amit Kemelmakher, Ariel McClendon, David Simons, Jason Fletcher and George W. Bush. We also thank Todd Bogdan for his help with the Picasa implementation. Pictures of George W. Bush are used with permission by Reuters (teaser Fig. photos 2,4, 8; Fig. 1; Fig. 10 (c) photos 1, 4, 5, 8, 10, 11, 12) and by AP Photo (teaser Fig. photos 1, 3, 5, 6, 7; Fig. 10 (c) photos 2, 3, 6, 7, 9).

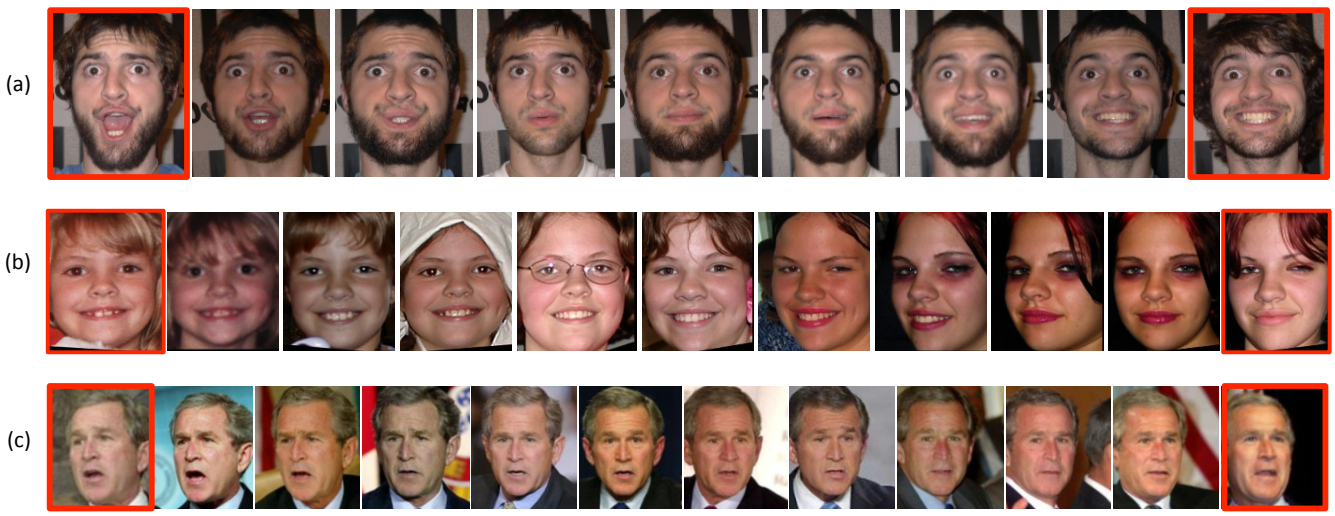


Figure 10: *Example paths produced with our method using different type of datasets (a) time-lapse photo collection of Jason (1598 photos), (b) personal photo collection of Ariel over 20 years (1300 photos) and (c) George W. Bush photo collection (530 photos). The end points (marked in red) were chosen by the user and all the intermediate pictures were selected automatically by our method. Note the smooth transition in facial expression as well as pose.*

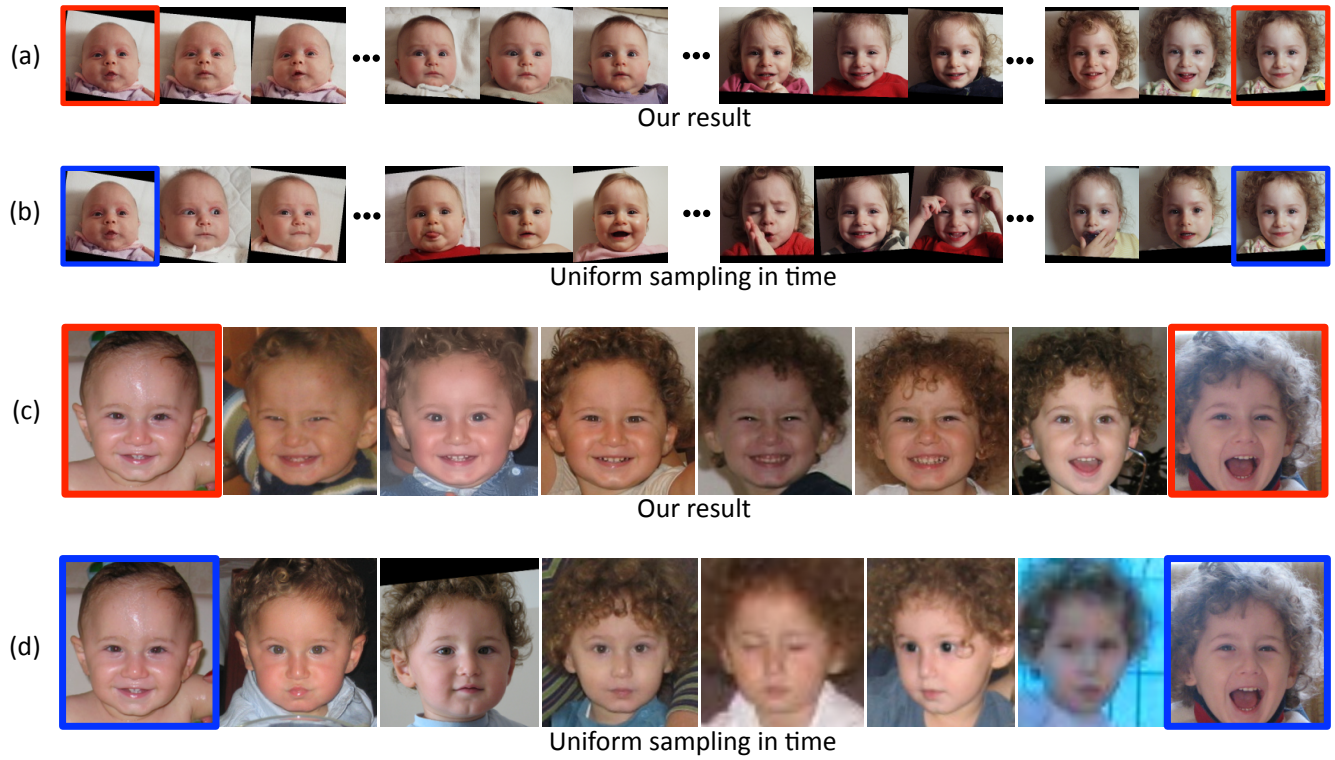


Figure 11: *Comparison between uniform sampling in time (no path optimization) and our result. When just sampling, highly non-coherent transitions occur, e.g, the face or mouth is occluded by hands, eyes close and open, large jumps in the mouth expression, etc.*

References

ADELSON, E. H., AND MOVSHON, J. A. 1982. Phenomenal coherence of moving visual patterns. *Nature 300*, 5892, 523–525.

AHONEN, T., HADID, A., AND PIETIKINEN, M. 2006. Face description with local binary pat- terns: Application to face recognition. In *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 28, 2037–2041.

ARIKAN, O., AND FORSYTH, D. A. 2002. Interactive motion generation from examples. *ACM Trans. Graph. 21*, 3, 483–490.

BEDERSON, B. B. 2001. Photomesa: a zoomable image browser using quantum treemaps and bubblemaps. In *UIST*, 71–80.

BEIER, T., AND NEELY, S. 1992. Feature-based image metamorphosis. 35–42.

BERG, T. L., BERG, A. C., EDWARDS, J., MAIRE, M., WHITE, R., TEH, Y.-W., LEARNED-MILLER, E., AND FORSYTH, D. A. 2004. Names and faces in the news. In *CVPR*, 848–854.

BITOUK, D., KUMAR, N., DHILLON, S., BELHUMEUR, P., AND NAYAR, S. K. 2008. Face swapping: automatically replacing faces in photographs. In *SIGGRAPH*, 1–8.

BLANZ, V., AND VETTER, T. 1999. A morphable model for the synthesis of 3d faces. In *SIGGRAPH*, 187–194.

BOURDEV, L., AND BRANDT, J. 2005. Robust object detection via soft cascade. *CVPR*.

BREGLER, C., COVELL, M., AND SLANEY, M. 1997. Video rewrite: Driving visual speech with audio. In *SIGGRAPH*, 75–84.

CHEN, S. E., AND WILLIAMS, L. 1993. View interpolation for image synthesis. In *SIGGRAPH*, 279–288.

DALAL, N., AND TRIGGS, B. 2005. Histograms of oriented gradients for human detection. In *CVPR*, 886–893.

EVERINGHAM, M., SIVIC, J., AND ZISSERMAN, A. 2006. "Hello! My name is... Buffy" – automatic naming of characters in TV video. In *Proceedings of the British Machine Vision Conference*.

FREEMAN, W. T., ADELSON, E. H., AND HEEGER, D. J. 1991. Motion without movement. *Computer Graphics 25*, 27–30.

GOLDMAN, D. B., GONTERMAN, C., CURLESS, B., SALESIN, D., AND SEITZ, S. M. 2008. Video object annotation, navigation, and composition. In *UIST*, 3–12.

GRAHAM, A., GARCIA-MOLINA, H., PAEPCKE, A., AND WINOGRAD, T. 2002. Time as essence for photo browsing through personal digital libraries. In *JCDL*, 326–335.

HUANG, G. B., RAMESH, M., BERG, T., , AND LEARNED-MILLER, E. 2007. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Tech. Rep. 07-49, University of Massachusetts, Amherst.

HUYNH, D. F., DRUCKER, S. M., BAUDISCH, P., AND WONG, C. 2005. Time quilt: scaling up zoomable photo browsers for large, unstructured photo collections. In *CHI*, 1937–1940.

JOSHI, N., SZELISKI, R., AND KRIEGMAN, D. J. 2008. Psf estimation using sharp edge prediction. In *CVPR*.

KATZ, S., TAL, A., AND BASRI, R. 2007. Direct visibility of point sets. *SIGGRAPH 26*, 3.

KEMELMACHER-SHLIZERMAN, I., SANKAR, A., SHECHTMAN, E., AND SEITZ, S. M. 2010. Being John Malkovich. *ECCV*.

KENKEL, F. 1913. Untersuchungenber den zusammenhang zwischen erscheinungsgrsse und ersehrinungsbewegung bei einigen sogenannten optischen tuschungen. *Z. Psychol. 67*, 358–449.

KOVAR, L., GLEICHER, M., AND PIGHIN, F. 2002. Motion graphs. In *SIGGRAPH*, 473–482.

KUMAR, N., BELHUMEUR, P., AND NAYAR, S. 2008. Facetracer: A search engine for large collections of images with faces. In *ECCV*, 340–353.

LASSETER, J. 1987. Principles of traditional animation applied to 3D computer animation. In *Proc. SIGGRAPH 87*, 35–44.

LEVOY, M., AND HANRAHAN, P. 1996. Light field rendering. In *SIGGRAPH*, 31–42.

LU, Z.-L., AND SPERLING, G. 2002. Three systems theory of human visual motion perception. *JOSA A 19*, 2, 413–413.

LUCAS, B., AND KANADE, T. 1981. An iterative image registration technique with an application to stereo vision. *Proceedings of Imaging Understanding Workshop*, 121–130.

MARR, D., AND HILDRETH, E. 1980. Theory of edge detection. *Proc. R. Soc. Lond. B 207*, 187–217.

NALWA, V. S., AND BINFORD, T. O. 1986. On detecting edges. *IEEE Trans. Pattern Anal. Mach. Intell. 8*, 699–714.

OJALA, T., PIETIKINEN, M., AND MENP, T. 2002. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, 971–987.

PENTLAND, A., PICARD, R. W., AND SCLAROFF, S. 1996. Photobook: Content-based manipulation of image databases. *International Journal of Computer Vision 18*, 3, 233–254.

PICASA, 2010. http://googlephotos.blogspot.com/2010/08/picasa-38-face-movies-picnik.html.

PIGHIN, F., HECKER, J., LISCHINSKI, D., SZELISKI, R., AND SALESIN, D. H. 1998. Synthesizing realistic facial expressions from photographs. In *SIGGRAPH*, 75–84.

REKIMOTO, J. 1999. Time-machine computing: a time-centric approach for the information environment. In *UIST*, 45–54.

SEITZ, S. M., AND DYER, C. R. 1996. View morphing. In *SIGGRAPH*, 21–30.

SHASHUA, A. 1992. *Geometry and Photometry in 3D Visual Recognition*. PhD thesis, Massachusetts Institute Of Technology, Cambridge, MA.

SNAVELY, N., GARG, R., SEITZ, S. M., AND SZELISKI, R. 2008. Finding paths through the world's photos. *ACM Trans. Graph. 27*, 3, 1–11.

SZELISKI, R., AND SHUM, H.-Y. 1997. Creating full view panoramic image mosaics and environment maps. In *Proc. SIGGRAPH 97*, 251–258.

WERTHEIMER, M. 1912. Experimentelle studien uber das sehen von bewegung. *Z. Psychol. 61*, 161–265.

ZHANG, L., SNAVELY, N., CURLESS, B., AND SEITZ, S. M. 2004. Spacetime faces: high resolution capture for modeling and animation. In *SIGGRAPH*, 548–558.