# Towards Space-Time Light Field Rendering

Huamin Wang[*]
Georgia Institute of Technology

Ruigang Yang[†]
University of Kentucky

## Abstract

So far extending light field rendering to dynamic scenes has been trivially treated as the rendering of static light fields stacked in time. This type of approaches requires input video sequences in strict synchronization and allows only discrete exploration in the temporal domain determined by the capture rate. In this paper we propose a novel framework, *space-time light field rendering*, which allows *continuous* exploration of a dynamic scene in both spatial and temporal domain with *unsynchronized* input video sequences.

In order to synthesize novel views from any viewpoint at any time instant, we develop a two-stage rendering algorithm. We first interpolate in the temporal domain to generate globally synchronized images using a robust spatial-temporal image registration algorithm followed by edge-preserving image morphing. We then interpolate those software-synchronized images in the spatial domain to synthesize the final view. Our experimental results show that our approach is robust and capable of maintaining photo-realistic results.

**CR Categories:** I.3.3 [Computer Graphics]: Picture/Image Generation—Display algorithms, Viewing algorithms; I.4.8 [Image Processing and Computer Vision]: Scene Analysis—Time-varying imagery

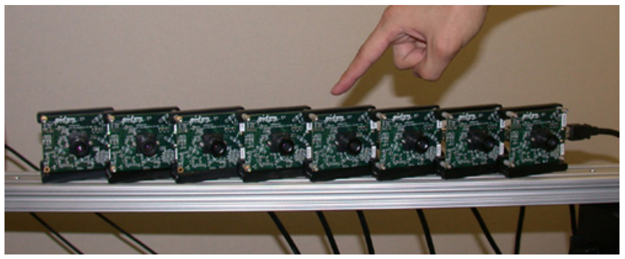
**Keywords:** image-based rendering, space-time light field, epipolar constraints

## 1 Introduction

During the last few years, image-based modeling and rendering (IBMR) has become a popular alternative of synthesizing photo-realistic images of real scenes, whose complexity and subtleties are difficult to capture with traditional geometry based techniques. Among the many IBMR approaches, light field rendering (LFR) [Levoy and Hanrahan 1996; Gortler et al. 1996] is a representative one, which uses many images captured from different view points of the scene of interests, i.e., a *light field*. Given the light field, the rendering process becomes a simple operation of rearranging the recorded pixel values. It can guarantee the rendering quality for any type of scenes, without resorting to relatively difficult and often fragile depth reconstruction approaches.

So far, extending the success of light field rendering to dynamic scenes has been limited to *synchronized* input video sequences. Each set of images taken at the same instant is treated as a separate light field. Although a viewer can continuously change the viewpoint in the spatial domain within each static light field, exploring in the temporal domain is still discrete.

[*]e-mail: whmin@cc.gatech.edu
[†]e-mail: ryang@cs.uky.edu



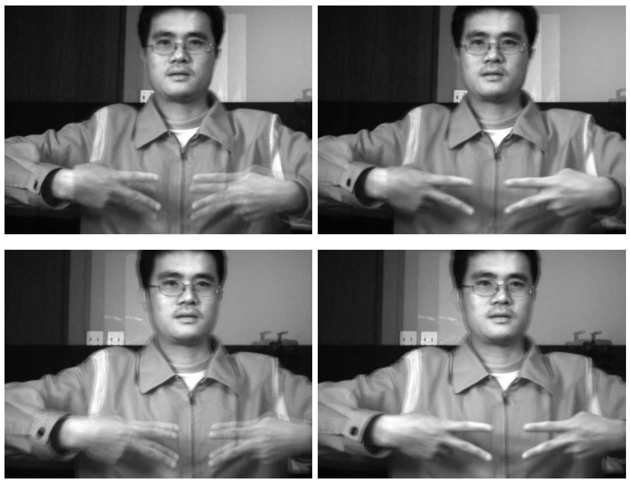


Figure 1: The top picture shows the camera array we used to capture video sequences. The cameras are running at 30 fps without inter-camera synchroization. The pictures in the left column are synthesized using traditional light field rendering. The pictures in the right column are synthesized using space-time light field rendering.

In this paper we present the notion of *space-time light field rendering* (ST-LFR) that allows *continuous* exploration of a dynamic light field in both the spatial and temporal domain. A space-time light field is defined as a collection of video sequences in which each frame has a global time stamp but may or may not be synchronized between sequences. The goal of ST-LFR is to synthesize novel images given an arbitrary viewpoint at an arbitrary time instant $t$. Traditional LFR is therefore a special case of ST-LFR with a fixed $t$. One may think that, given a high-enough capture rate (i.e., full NTSC rate at 30 frames per second), applying traditional LFR to the set of images captured most closely to $t$ could generate reasonable results. Unfortunately, a back-the-envelop calculation shows that this is usually not true. Typical body motions (approximately 0.6 m/s) at sitting distance can cause over a 10-pixel offset per frame in a 30fps VGA sequence. As shown in Figure 1 left column, the misalignment caused by the subject's casual hand motion is quite evident in the synthesized image even when the temporal offset is less than a half frame time, i.e, $1/60$ second.

We decompose ST-LFR into two stages. First we interpolate original video sequences in the temporal domain to synthesize images for a given time instant. Given these virtually synchronized images, we use the traditional LFR technique to render novel views in the spatial domain. The major technical difficulty of ST-LFR is in the first step: synthesizing 'in-between' frames within each sequence. Towards this end, we made two primary **contributions**. First we developed a robust spatial-temporal image registration algorithm using optical flow and epipolar constraints. The number of mismatches can be greatly reduced after enforcing both inter-sequence and intra-sequence matching consistency. Our second contribution is a novel temporal image interpolation algorithm that provides superior results in the presence of inevitable mismatches. Our algorithm, based on image morphing [Beier and Neely 1992], incorporates a varying weight term based on image gradient to preserve edges.

Figure 1 shows the effectiveness of our approach. Though we have not demonstrated, ST-LFR can also be easily extended to increase video frame rates, visualize motion trajectory in 3D (i.e., integration in the temporal domain), or produce image-based motion blur effects as in [Brostow and Essa 2001].

## 2 Related Work

Light field rendering techniques are formulated around the *plenoptic function*, which describes all of the radiant energy that can be perceived by an observer at any point in space and time [Adelson and Bergen 1991]. Levoy and Hanrahan pointed out that the plenoptic function can be simplified in the regions of space free of occluders, since radiant energy does not change over free space. The reduced function (the so-called *light field*) can be interpreted as a function on the space of oriented light rays. The light field can be directly recorded using multiple cameras. Once acquired, light field rendering becomes a simple task of table–look–ups to retrieve the recorded radiant values.

A major disadvantage of light field based techniques is the huge amount of data required, therefore their application was traditionally limited static scenes [Levoy and Hanrahan 1996; Gortler et al. 1996; Shum and He 1997; Ihm et al. 1998]. The acquisition of dense, dynamic light fields becomes feasible only recently with technological advancement. Most systems use a dense array of *synchronized* cameras to acquire dynamic scenes at discreet temporal intervals [Naemura et al. ; Yang et al. 2002a; Wilburn et al. 2002; Matusik and Pfister 2004]. Images at each time instant are collectively treated as an independent light field. As a result, viewers can explore continuously only in the spatial domain, but not in the temporal domain. To create the impression of dynamic events, the capture frame rate has to be sufficiently high, further increasing the demand for data storage and processing. In addition, hardware camera synchronization is a feature available only on high-end cameras, which significantly increases the system cost. Without synchronization,, misalignment problems will occur in fast motion scenarios, as reported in one of the few light field systems built with low cost commodity web cameras [Zhang and Chen 2004].

In the meantime, view synthesis from a *sparse* set of cameras is also an active research topic in computer vision. Many of these techniques focus on the dense geometric reconstruction of the scene (e.g. [Kanade et al. 1997; Matusik et al. 2000; Yang et al. 2002b; Zitnick et al. 2004]). While some very impressive results have been obtained, the problem of depth reconstruction remains open. In addition, synchronized camera input is typically required. With unsynchronized input, there are techniques to find the temporal offset (typically at a resolution of one frame) [Caspi and Irani 2000; Caspi and Irani 2001; Rao et al. 2003; Sand and Teller 2004], but they do not address the problem of synthesizing the 'in-between' frames, which is a critical part of ST-LFR.

Our work is also related to super-resolution techniques in computer vision. While most approaches focus on increasing the spatial resolution (See [Borman and Stevenson 1998] for a comprehensive review), a notable exception in [Shechtman et al. 2002] extends the notion of super-resolution to the space-time domain. Their approach enables new visual capabilities of dynamic events such as the removal of motion blur and temporal anti-aliasing. However, they assume that all images have already been registered in the space and time volume and the scene is approximately planar. Our ST-LFR formulation does not require a priori image registration and does allow arbitrary scene and motion types. On the other hand, we do not attempt to increase the spatial resolution.

The key component of ST-LFR is image registration, which is achieved through optical flow computation over both space and time. Optical flow is a well-studied problem in computer vision. Interested readers are referred to [Bergen et al. 1992] and [Barron et al. 1994] for reviews and comparison of different optical flow techniques. These flow techniques typically use only a single sequence of frames (two frames in most cases). In our case we have many sequences taken simultaneously, therefore, we extend the optical flow formulation in both the space and time domain by incorporating the powerful epipolar constraint.

## 3 Algorithm Overview

The problem we address is to synthesize images from an arbitrary viewpoint at an arbitrary time instant, given a set of time-stamped video sequences captured from known positions, i.e., a *space-time light field*. Assuming that the video sequences are not synchronized, our approach to space-time light field rendering (ST-LFR) is divided into three major steps as shown in Figure 2:

1. Image registration
2. Synchronized image generation (temporal interpolation)
3. Light field rendering (spatial interpolation)

In the first step (Section 4), we use a robust spatial-temporal optical flow algorithm to establish feature correspondences among successive frames for each camera's sequence. In the second step (Section 5), new globally synchronized images are synthesized using a novel edge-guided image morphing method. Synchronized video sequences are eventually used as input to synthesize images in the spatial domain using the traditional LFR technique (Section 6).

## 4 Image Registration

Our algorithm starts with computing the temporal optical flow between two successive frames for each camera. The spatial optical flow between different camera sequences is also computed to remove outliers. We will first present our algorithm with a two-camera setup, then show how it can be extended to handle multiple camera inputs. Figure 3 illustrates the two-camera registration process.

### 4.1 Spatial-Temporal Flow Computation

Let $I_{i,t}$ and $I_{i,t+\Delta t_i}$ be two successive frames captured from camera $C_i$ at time $t$ and $t+\Delta t_i$ respectively, and $I_{j,t'}$ be captured from camera $C_j$ at time $t'$, the temporal flow is defined from $I_{i,t}$ to $I_{i,t+\Delta t_i}$ and the spatial flow is from $I_{i,t}$ to $I_{j,t'}$. Camera $C_i$ and $C_j$'s frame time steps ($\Delta t_i$ and $\Delta t_j$) are not necessarily equivalent.

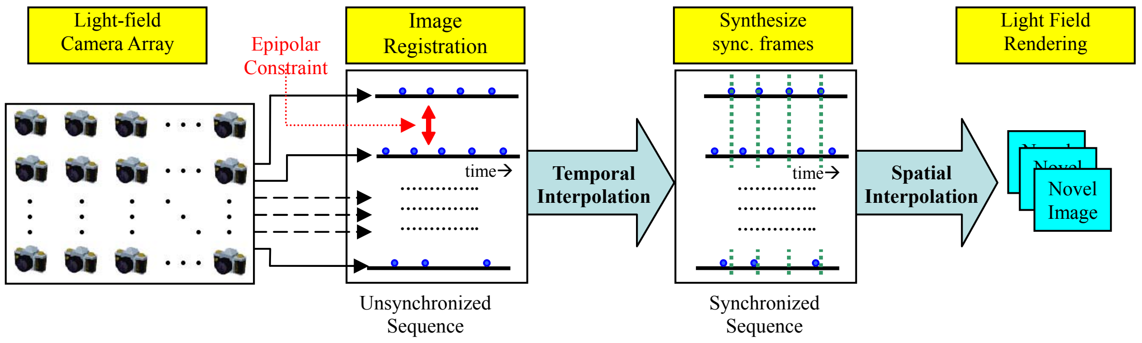


Figure 2: The structure of our space-time light field rendering algorithm.

We first select corners with large eigenvalues on $I_{i,t}$ as feature points using Harris corner detector [Harris and Stephens 1988]. We enforce a minimum distance between any two feature points to prevent them from gathering in a small high gradient region. We then calculate the sparse optical flow using tracking functions in OpenCV [Bouguet 1999], which is based on the classic Kanade-Lucas-Tomasi(KLT) feature tracker [Lucas and Kanade 1981; Tomasi and Kanade 1991].

We choose not to calculate a dense, per-pixel flow since there are certain areas such as occlusion boundaries and textureless regions where flow computation is known to be problematic. Our sparse flow formulation, which includes salient features important for view synthesis, is not only more robust, but also more computationally efficient.

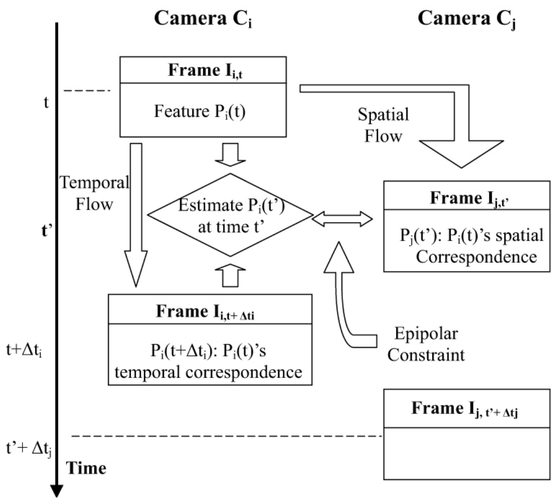


Figure 3: The registration process of the two-camera case.

## 4.2 Flow Correction by Epipolar Constraint

Since image interpolation quality depends heavily on the accuracy of feature point correspondences, we use the *epipolar constraint* to detect temporal point correspondence errors.

Given two images captured at the same instant from two different cameras, the epipolar constraint states that if a point $\mathbf{p} = [u, v, 1]^T$ (expressed in homogeneous coordinates) from one camera and a point $\mathbf{q} = [u', v', 1]^T$ from another camera correspond to the same stationary 3D point $\mathbf{m}$ in the physical world, they must satisfy the following equation:

$$\mathbf{q}^T \mathbf{F} \mathbf{p} = 0 \tag{1}$$

where $\mathbf{F}$ is the fundamental matrix encoding the epipolar geometry between the two images [Faugeras 1993]. In fact, $\mathbf{Fp}$ defines the epipolar line in the second image, thus Eq. 1 means that the point $\mathbf{q}$ must pass through the epipolar line $\mathbf{Fp}$, and vice versa.

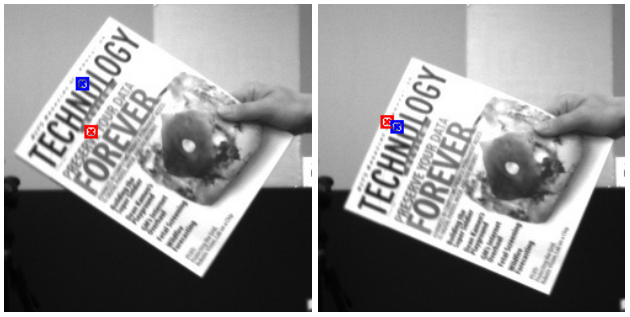
We incorporate the epipolar constraint to verify the temporal flow from $I_{i,t}$ to $I_{i,t+\Delta t_i}$ using the spatial flow from $I_{i,t}$ to $I_{j,t'}$. Let $\mathbf{m}(t)$ be a moving 3D point at time $t$ and $\mathbf{p_i}(t)$ be $\mathbf{m}(t)$'s projection in $C_i$. Given two 2D image points in different cameras, $\mathbf{p_i}(t_1)$ and $\mathbf{p_j}(t_2)$, we claim that they satisfy the epipolar constraint if $t_1 = t_2$ and $\mathbf{p_i}(t_1)\mathbf{F_{ij}}\mathbf{p_j}(t_2) = 0$ where $\mathbf{F_{ij}}$ is the fundamental matrix between camera $C_i$ and $C_j$. We first estimate the image point $\mathbf{p_i}(t')$ using our unsynchronized images $I_{i,t}$ and $I_{i,t+\Delta t_i}$. Assuming locally linear motion, $\mathbf{p_i}(t')$ can be estimated as:

$$\mathbf{p_i}(t') = (t + \Delta t_i - t') \cdot \mathbf{p_i}(t) + (t' - t) \cdot \mathbf{p_i}(t + \Delta t_i). \tag{2}$$

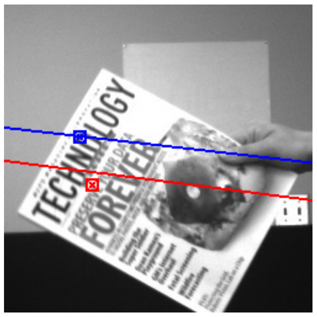
If $\mathbf{p_i}(t)$'s spatial and temporal correspondences are both correct, $\mathbf{p_i}(t')$ and $\mathbf{p_j}(t')$ should satisfy the epipolar constraint. We use this criterion to validate the spatial-temporal flow computation. Figure 4 shows that a correct correspondence (blue) satisfies the epipolar constraint, while a wrong temporal correspondences (red) causes an error in $\mathbf{p_i}(t')$, which leads to a wrong epipolar line that $\mathbf{p_j}(t')$ fails to meet with.

We calculate $\mathbf{p_i}(t')$'s epipolar line on $I_{j,t'}$ using the fundamental matrix computed from cameras' world position and projection matrices. Due to various error sources such as camera noise, inaccuracy in camera calibration or feature localization, we define a band of certainty along the epipolar line. For every triplet $(\mathbf{p_i}(t'), \mathbf{p_j}(\mathbf{t'}), \mathbf{m}(t'))$, if the distance from $\mathbf{p_j}(t')$ to $\mathbf{p_i}(t')$'s epipolar line is greater than a certain tolerance threshold, either the temporal or the spatial flow for this triplet is assumed to be wrong. This feature will be discarded. In our experiment, we use three pixels as the distance threshold.

It should be noted that our correction scheme for unsynchronized input is only valid when the motion is roughly linear in the projective space. Many real world movements, such as rotation, do not satisfy this requirement. Fortunately, when cameras have a sufficiently high rate with respect to the 3D motion, such a locally temporal linearization is generally acceptable [Zhang et al. 2003]. Our experimental results also support this assumption. In Figure 4, correct feature correspondences satisfy the epipolar constraint well even though the magazine was rotating fast. Figure 4 also demonstrates the amount of pixel offset casual motion could introduce: in two successive frames captured at 30fps, many feature points



(a) $\mathbf{p_i}(t)$ on image $I_{i,t}$ (b) $\mathbf{p_i}(t+\Delta t_i)$ on image $I_{i,t+\Delta t_i}$



(c) $\mathbf{p_j}(t')$ and $\mathbf{p_i}(t')$'s epipolar lines on image $I_{j,t'}$

Figure 4: Feature points and epipolar line constraints.

moved more than 20 pixels—a substantial amount that cannot be ignored in view synthesis.

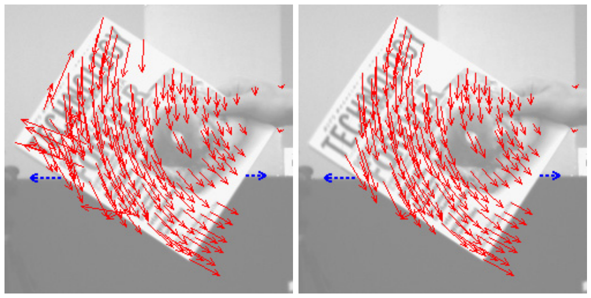


Figure 5: The optical flow before (left image) and after (right image) applying the epipolar constraint. Correspondence errors on the top of the magazine are detected and removed.

### 4.3 Multi-camera Flow Correction

In a multi-camera system, we can use more than one reference camera or multiple frames to justify a particular temporal flow using the epipolar constraint. Since the spatial flow itself can contain errors, there exists a trade-off between accurate temporal correspondences and the number of false-negatives, i.e., correct temporal correspondences are removed because of erroneous spatial correspondences. Therefore, we need a selection scheme to choose the 'best' frame(s) to compare. Intuitively, closer cameras are relatively good candidates because of less occlusions.

Another factor to consider is the ambiguity along the epipolar line, i.e., the correspondence error along the direction of the epipolar line cannot be detected. Many light-field acquisition systems put cameras on a regular grid, in which the epipolar lines for cameras in the same row or column are almost aligned with each other. In this case, using more reference cameras does not necessarily improve error detection.

Given a reference camera, we prefer to choose reference frames captured close to time $t+\Delta t_i$. Those frames can fully reveal the temporal flow error since the error only exists in $\mathbf{p_i}(t)$'s temporal correspondence $\mathbf{p_i}(t+\Delta t_i)$. $\mathbf{p_i}(t)$ is always assumed to be the correct image location of some 3D point $\mathbf{m}(t)$.

We use a selection function $W_j$ to determine whether camera $C_j$ should be used to test $C_i$'s temporal flow from $I_{i,t}$ to $I_{i,t+\Delta t_i}$:

$$W_j = Closeness(i,j) + d \min_{t':I_{j,t'} \in C_j \ sequence} \left| t' - (t+\Delta t_i) \right| \tag{3}$$

and the reference frame $I_{j,t'}$ from $C_j$ is selected as:

$$t' = \arg\min_{t':I_{j,t'} \in C_j \ sequence} \left| t' - (t+\Delta t_i) \right| \tag{4}$$

where $d$ is a constant to balance the influence from the camera spatial closeness and the capture time difference. If the multi-camera system is constructed regularly as a camera array, the closeness can be simply evaluated according to the array indices. We choose a single best camera along the row and the column respectively to provide both horizontal and vertical epipolar constraints. If all cameras have an identical frame rate, the same camera will always be selected using Eq. 3.

## 5 Edge-Guided Temporal Interpolation

After image registration, we obtain a set of matching feature points between consecutive frames. We next interpolate frames temporally to generate synchronized video sequences. One possible approach is to simply triangulate the whole image and blend two images using texture mapping. However, the result is unpredictable since it depends heavily on the local triangulation of feature points. Even a single feature mismatch can affect a large region on the image.

Our temporal interpolation scheme, which is based on the image morphing method ([Beier and Neely 1992]), incorporates varying weights based on image gradient. In essence, we form lines for image morphing by examine feature points pairwise. The lines that are aligned with image edges gain extra weights to preserve edge straightness – an important visual cue. Our method can synthesize smooth and visually appealing images even in the presence of missing or wrong feature matches.

### 5.1 Edge detection and matching

We first examine feature point correspondences pairwise to find potential feature edges. Let $E_{i,t}$ be an edge segment connected by two feature points in image $I_{i,t}$, and $E_{i,t+\Delta t_i}$ be the corresponding edge in image $I_{i,t+\Delta t_i}$. We choose corresponding samples on $E_{i,t}$ and $E_{i,t+\Delta t_i}$ from the image gradient map, and calculate a measure of fitness $c(x,y)$ for each sample at $(x,y)$:

$$c(x,y) = |\nabla I(x,y)| + \alpha \left| \arctan\left( \frac{\nabla I(x,y)_y}{\nabla I(x,y)_x} \right) - \psi_\perp \right| \tag{5}$$

where $\nabla I(x,y)$ is the gradient vector (subscript for image index dropped for simplicity), and $\psi_\perp$ is the angle between the edge normal (perpendicular to the edge) and the horizontal.

The first term in Eq. 5 is the gradient magnitude and the second term indicates how the gradient vector matches the edge direction. $\alpha$ is a parameter to balance the influence from two terms. A typical value for $\alpha$ is from 5 to 10, assuming that image intensities are in [0, 255] and the angles are measured in radians. If all fitness measures are greater than some threshold, it means $E_{i,t}$ and $E_{i,t+\Delta t_i}$ stay on a strong gradient region and the gradient direction matches with the edge normal. Thus, $E_{i,t}$ and $E_{i,t+\Delta t_i}$ may represent a real edge in the 3D world and they should be added into the feature edge set. Figure 6 shows a gradient magnitude map and detected feature edges.

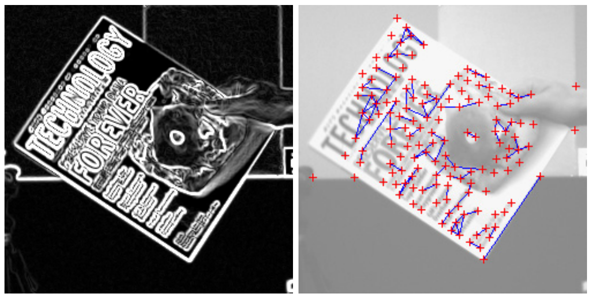


Figure 6: Real feature edges. Left: the gradient magnitude map. Right: feature edges (blue) detected by testing feature points pairwise.

**Edge Length Constraint** When the object deformation and motion is relatively slow, we assume that the feature edge length is roughly a constant between two successive frames. This assumption is used as a constraint to detect improper feature edge correspondences.

Wrong point correspondence is likely to cause the length change between corresponding edges because it is inconsistent with correct point correspondence in the temporal flow. For instance, in Figure 5, the dashed blue arrows show the intersection boundaries between the magazine and the wall. They do not represent real 3D points, neither do they follow the magazine motion. Removing edges ended with such feature points can avoid distortions during temporal interpolation. Similarly, we do not consider segments connected between one static point (i.e., a point that does not move in two temporal frames) and one dynamic point due to the segment length change.

To avoid unnecessary edge tests, we assume that maximum edge length is bounded by a constant $L_{max}$. For each feature point $P_0$, edge tests can then be limited only to those feature points in a circle centered at $P_0$ with radius $L_{max}$. A k-d tree structure is constructed to find feature point neighborhoods quickly. If $L_{max}$ is relatively small compared with the frame resolution, the computational cost will be greatly reduced from O($n^2$) to O(m·n), where $n$ is the total number of feature points and $m$ is the maximum number of feature points clustered in any circle with radius $L_{max}$.

## 5.2 Forming feature edge set

Usually the number of feature edges found so far is not enough to cover the whole image range. To avoid possible distortions in areas lacking detected features, we use constrained Delaunay triangulation [Chew 1987] to triangulate the whole image range and add new triangle edges into the feature edge set as *virtual edges*. Real feature edges detected using the technique in the previous section are used as triangulation constraints. Feature points and four image corners are used as triangle vertices. Note that we break intersecting feature edges since no intersection is allowed among edge constraints for triangulation. Figure 7 shows the triangulation result.

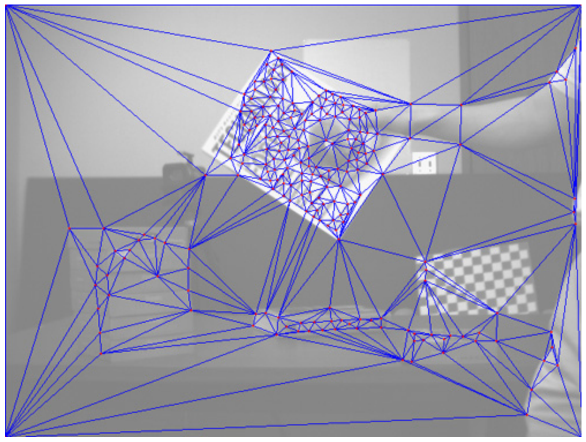


Figure 7: The edge map constructed after constrained Delaunay triangulation.

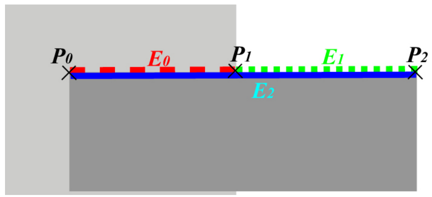


Figure 8: Three feature points $P_0$, $P_1$ and $P_2$ are selected on the same dark boundary, forming three edges $E_0$, $E_1$ and $E_2$. We only keep the longest one $E_2$, which represents the whole edge in the real world.

Sometimes, more than two feature points may be selected on a thick and high-gradient band, so that more than one feature edges may be generated (as shown in Figure 8) even though the band represents the same edge in the world space. This is problematic since those extra feature edges will repetitively apply more influence weight than they should during the next image morphing step. We detect those cases by testing whether two real feature edges are nearly parallel and close. When detected, we only keep the longer one.

## 5.3 Morphing

We extend the image morphing method [Beier and Neely 1992] to interpolate two frames temporally. We allow real edges to have more influence weight than virtual edges since they are believed to be more accurate. In [Beier and Neely 1992], edge weights are calculated using the edge length and the point-edge distance:

$$weight_0 = \left(\frac{length^p}{(a+dist)}\right)^b \tag{6}$$

where $a, b$ and $p$ are constants to change the line influence.

We calculate the weight for virtual feature edge using the formula above. The weight for real edge is boosted as:

$$weight = weight_0 \cdot (1 + \frac{\overline{c(x,y)} - \overline{c(x,y)}_{\min}}{\overline{c(x,y)}_{\max} - \overline{c(x,y)}_{\min}})^e \tag{7}$$

where $\overline{c(x,y)}$ is the edge samples' average fitness value (Section 5.1). $\overline{c(x,y)}_{\min}$ and $\overline{c(x,y)}_{\max}$ are the minimum and maximum $\overline{c(x,y)}$ of all real feature edges, respectively. $e$ is a parameter to scale the boosting effect exponentially. In practice $e$ is chosen from 1.2 to 2.

We temporally interpolate two frames using both the forward temporal flow from $I_{i,t}$ to $I_{i,t+\Delta t_i}$ and backward temporal flow from $I_{i,t+\Delta t_i}$ to $I_{i,t}$. The final pixel is calculated by linear interpolation:

$$P_{t'} = P_{forward} \cdot (t + \Delta t_i - t') + P_{backward} \cdot (t' - t) \qquad (8)$$

where $P_{forward}$ only uses frame $I_{i,t}$ and $P_{backward}$ only uses frame $I_{i,t+\Delta t_i}$. This is because we have more confidence with $I_{i,t}$'s features in the forward flow and $I_{i,t+\Delta t_i}$'s features in the backward flow. They are picked immediately from images using feature detection.

Figure 9 shows the results without and with virtual edges. Figure 10 shows the results using different interpolation schemes. Since some features are missing on the top of the magazine, the interpolation quality improves when real feature edges get extra weights according to Eq. 7.

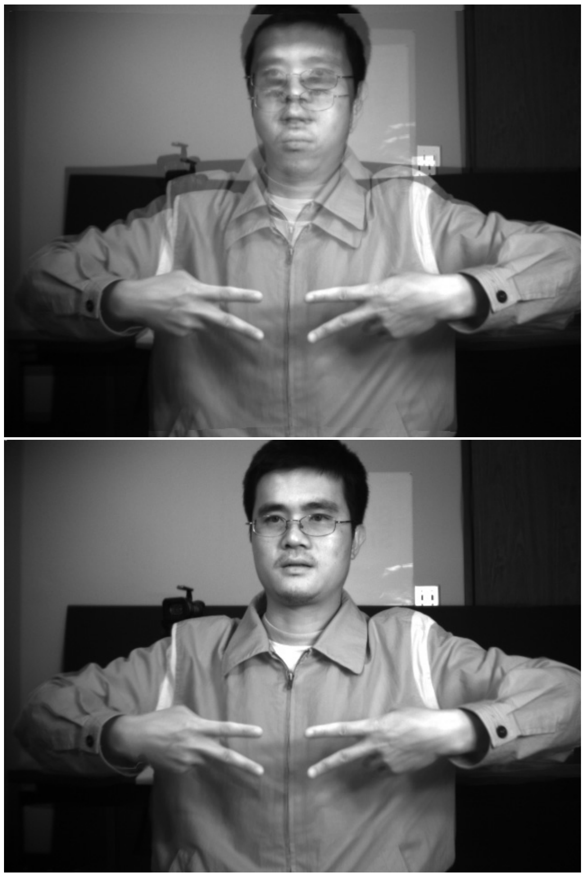


Figure 9: Interpolation result without (top image) and with (bottom image) virtual feature edges.

## 6 View Synthesis (Spatial Interpolation)

Once we generate synchronized images for a given time constant, we can use traditional light field rendering techniques to synthesize views from novel viewpoint. To this end, we adopted the unstructured lumigraph rendering (ULR) technique [Buehler et al. 2001], which can utilize graphics texture hardware to blend appropriate pixels together from nearest cameras in order to compose the desired image.

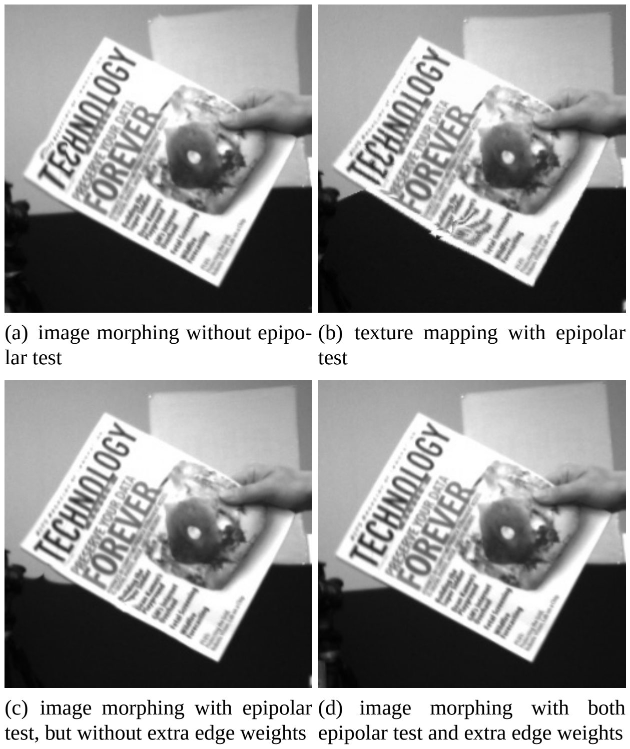

(a) image morphing without epipolar test (b) texture mapping with epipolar test (c) image morphing with epipolar test, but without extra edge weights (d) image morphing with both epipolar test and extra edge weights

Figure 10: The interpolated result using different schemes.

ULR requires an approximation of the scene (a geometric proxy) as an input. In our current implementation, the geometric proxy is a 3D plane that can be interactively controlled by the user. Its placement determines which region of the scene is in focus. It is also possible to use the spatial flow information to reconstruct a better proxy. However we chose not to do so in our experiments to better demonstrate the advantage of ST-LFR over traditional LFR (proxy computation is not a part of ULR).

## 7 Results

We have implemented our space-time light field rendering framework and tested with real data. In Figure 11 we show a graphical representation of our entire space-time light field rendering pipeline. To facilitate data capturing, we built a multi-camera system using eight Point Grey dragonfly cameras [Point Grey Research Inc. ]: three color ones and five grey-scale ones (see Figure 1 top). The mixed use of different cameras is due to our resource limit. These cameras are approximately 60 mm apart, limited by their form factor. Based on the analysis from [Chai et al. 2000], the effective depth of field is about 400mm. Four workstations are used to handle video stream capturing and one workstation is used to send operation instructions. Each video workstation captures video sequences from two cameras and store them as jpeg files. The global time stamp for each frame is available from Point Grey camera API. The cameras are calibrated, and all images are rectified to remove lens distortions.

Since our cameras are arranged along the horizontal, we only select one camera to use epipolar constraint according to Eq. 3. The closeness is just the camera position difference. In this case, we typically choose $d$ to be 60 in Eq. 3.

Our first data set includes a person waving the hand as Figure 1 shows. Since the depth variation from the hands to the head is slightly beyond the focus range (400mm), we can see some slight horizontal ghosting effects in the synthesized image, which are entirely different from vertical mismatches caused by the hand's vertical motion.

We emphasize the importance of virtual synchronization in Figure 12, in which the image is synthesized with a constant blending weight ($1/8$) for all eight cameras. The scene contains a moving magazine. Without virtual synchronization (left image), the text on the magazine cover is illegible. This problem is rectified after we registered feature points and generated virtually synchronized image frames.

Our last data set is a moving open book. In Figure 13 we synthesized several views using traditional LFR and ST-LFR. Results from ST-LFR remain sharp from different viewpoints. The noticeable change of intensity is due to the mixed use of color and greyscale cameras.

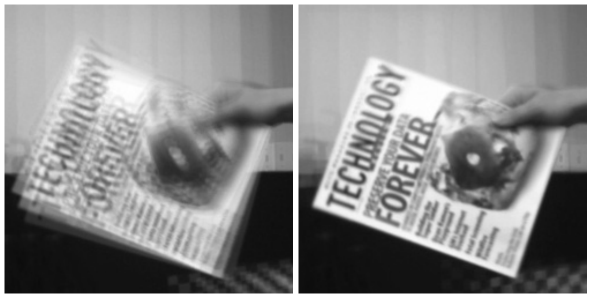


Figure 12: Synthesized results using a constant blending weight (i.e., pixels from all frames are averaged together). Left: traditional LFR with unsynchronized frames. Right: Space-time LFR.

Compared to the traditional light field rendering, our formulation requires correspondence information. Finding correspondence is an open problem in computer vision and it can be fragile in practice. While we have designed several robust techniques to handle almost inevitable mismatches and experimental results have been quite encouraging, we have not studied our techniques' stability over long sequences yet. On the other hand, the biggest problem in matching–textureless regions [Baker et al. 2003]–is circumvented in our formulation since we only compute a sparse flow field that is sufficient to synthesize novel views.

Regarding speed, our current implementation is not real-time yet. The average time for synthesizing one single frame takes a few minutes, depending on the number of detected features. The main bottleneck is the temporal interpolation step. To form the matching edge set also requires a quite exhaustive search, even after some acceleration structures are adopted.

## 8 Conclusion and Future Work

In this paper we extend the traditional light field rendering into the temporal domain to accommodate dynamic scenes. Instead of capturing the dynamic scene in strict synchronization and treating each image set as an independent static light field, our notion of a *space-time light field* simply assumes a collection of time-stamped video sequences. These sequences may or may not be synchronized and they can have different capture rates.

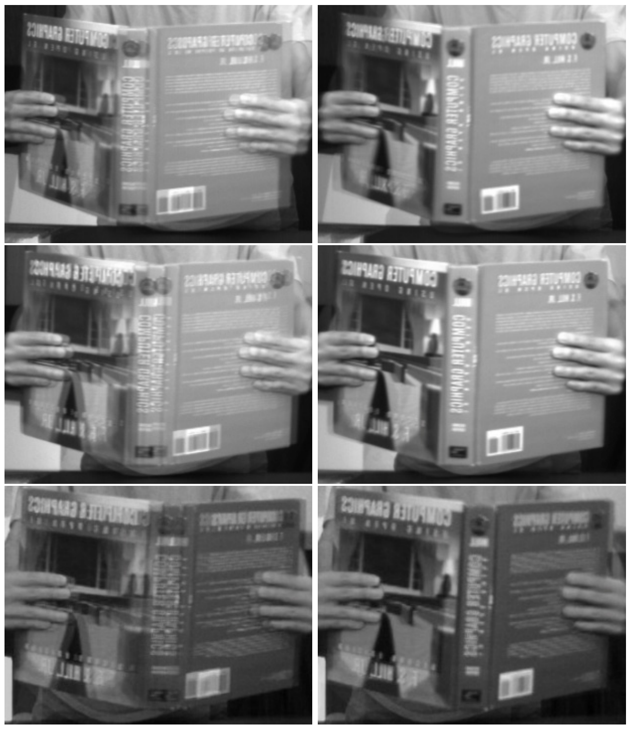


Figure 13: Synthesized results of the book sequence. Left: traditional LFR with unsynchronized frames. Right: Space-time LFR.

In order to be able to synthesize novel views from any viewpoint at any time instant, we developed a two-stage rendering algorithm: (1) temporal interpolation that registers successive frames with spatial-temporal optical flow and generates synchronized frames using an edge-guided image morphing algorithm that preserves important edge features; and (2) spatial interpolation that uses unstructured lumigraph rendering to create the final view from a given viewpoint. Experimental results have shown that our approach is robust and capable of maintaining photo-realistic results comparable to traditional static light field rendering.

Looking into the future, we intend to combine the temporal and spatial interpolation into a single step. By allowing blending over the entire space-time light field space, we can produce novel visual effects, such as the visualization of motion trajectory in 3D or motion blur. We also want to investigate more efficient interpolation method so that we can render space-time light field in real-time and online, without significantly affecting the image qualities. This real-time capability, together with the use of inexpensive web cameras, can contribute to a wider diffusion of light field techniques to many interesting applications such as 3D video teleconferencing, remote surveillance, and tele-medicine.

## References

ADELSON, E. H., AND BERGEN, J. 1991. The Plenoptic Function and the Elements of Early Vision. In *Computational Models of Visual Processing*, MIT Press, Cambridge, MA, 320.

BAKER, S., SIM, T., AND KANADE, T. 2003. When is the shape of a scene unique given its light-field: A fundamental theorem of 3d vision? *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI) 25*, 1, 100 – 109.

BARRON, J., FLEET, D. J., AND BEAUCHEMIN, S. S. 1994. Performance of Optical Flow Techniques. *International Journal of Computer Vision 12*, 1, 43–77.

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

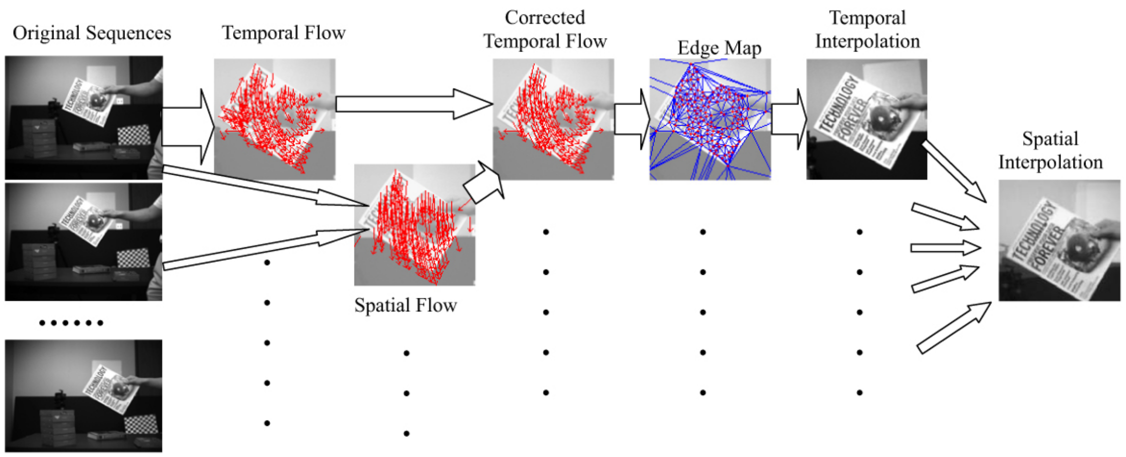


Figure 11: The space-time light field rendering pipeline.

BERGEN, J. R., ANANDAN, P., HANNA, K. J., AND HINGORANI, R. 1992. Hierarchical Model-Based Motion Estimation. In *Proceedings of the Second European Conference on Computer Vision (ECCV)*, 237–252.

BORMAN, S., AND STEVENSON, R. 1998. Spatial resolution enhancement of low-resolution image sequences - a comprehensive review with directions for future research. Technical report, Laboratory for Image and Signal Analysis (LISA), University of Notre Dame.

BOUGUET, J.-Y. 1999. Pyramidal implementation of the lucas kanade feature tracker description of the algorithm.

BROSTOW, G. J., AND ESSA, I. 2001. Image-based motion blur for stop motion animation. In *Proceedings of SIGGRAPH 2001*, ACM Press / ACM SIGGRAPH, E. Fiume, Ed., Computer Graphics Proceedings, Annual Conference Series, ACM, 561–566.

BUEHLER, C., BOSSE, M., MCMILLAN, L., GORTLER, S., AND COHEN, M. 2001. Unstructured Lumigraph Rendering. In *Proceedings of SIGGRAPH 2001*, 405–432.

CASPI, Y., AND IRANI, M. 2000. A Step towards Sequence to Sequence Alignment. In *Proceedings of Conference on Computer Vision and Pattern Recognition (CVPR)*, 682C689.

CASPI, Y., AND IRANI, M. 2001. Alignment of Non-Overlapping Sequences. In *Proceedings of International Conference on Computer Vision (ICCV)*, 76–83.

CHAI, J.-X., TONG, X., CHAN, S.-C., AND SHUM, H.-Y. 2000. Plenoptic Sampling. In *Proceedings of SIGGRAPH 2000*, 307318.

CHEW, L. P. 1987. Constrained delaunay triangulations. In *Proceedings of the third annual symposium on Computational geometry*, 215–222.

FAUGERAS, O. 1993. *Three-Dimensional Computer Vision: A Geometric Viewpoint*. MIT Press.

GORTLER, S. J., GRZESZCZUK, R., SZELISKI, R., AND COHEN, M. F. 1996. The Lumigraph. In *Proceedings of SIGGRAPH 1996*, 43–54.

HARRIS, C. J., AND STEPHENS, M. 1988. A combined corner and edge detector. In *Proceedings of 4th Alvey Vision Conference*, 147–151.

IHM, I., PARK, S., AND LEE, R. K. 1998. Rendering of Spherical Light Fields. In *Pacific Graphics*.

KANADE, T., RANDER, P. W., AND NARAYANAN, P. J. 1997. Virtualized Reality: Constructing Virtual Worlds from Real Scenes. *IEEE MultiMedia Magazine 1*, 1, 34–47.

LEVOY, M., AND HANRAHAN, P. 1996. Light Field Rendering. In *Proceedings of SIGGRAPH 1996*, 31–42.

LUCAS, B. D., AND KANADE, T. 1981. An Iterative Image Registration Technique with an Application to Stereo Vision. In *Proceedings of International Joint Conference on Artificial Intelligence*, 674–679.

MATUSIK, W., AND PFISTER, H. 2004. 3D TV: a Scalable System for Real-time Acquisition, Transmission, and Autostereoscopic Display of Dynamic Scenes. *ACM Trans. Graph. (SIGGRAPH Proceeding) 23*, 3, 814–824.

MATUSIK, W., BUEHLER, C., RASKAR, R., GORTLER, S., AND MCMILLAN, L. 2000. Image-Based Visual Hulls. In *Proceedings of SIGGRAPH 2000*, 369–374.

NAEMURA, T., TAGO, J., AND HARASHIMA, H. Realtime video-based modeling and rendering of 3d scenes.

POINT GREY RESEARCH INC. http://www.ptgrey.com.

RAO, C., GRITAI, A., SHAH, M., AND SYEDA-MAHMOOD, T. 2003. View-invariant Alignment and Matching of Video Sequences. In *Proceedings of International Conference on Computer Vision (ICCV)*, 939–945.

SAND, P., AND TELLER, S. 2004. Video matching. *ACM Transactions on Graphics 23*, 3, 592–599.

SHECHTMAN, E., CASPI, Y., AND IRANI, M. 2002. Increasing Space-Time Resolution in Video. In *Proceedings of the 7th European Conference on Computer Vision(ECCV)*, 753–768.

SHUM, H. Y., AND HE, L. W. 1997. Rendering with Concentric Mosaics. In *Proceedings of SIGGRAPH 1997*, 299–306.

TOMASI, C., AND KANADE, T. 1991. Detection and tracking of point features. Technical Report CMU-CS-91-132, Carnegie Mellon University, April.

WILBURN, B., SMULSKI, M., LEE, H., AND HOROWITZ, M. 2002. The Light Field Video Camera. In *Proc. Media Processors 2002, SPIE Electronic Imaging 2002*.

YANG, J. C., EVERETT, M., BUEHLER, C., AND MCMILLAN, L. 2002. A Real-time Distributed Light Field Camera. In *In Proceedings of the 13th Eurographics Workshop on Rendering*, 77–86.

YANG, R., WELCH, G., AND BISOP, G. 2002. Real-Time Consensus-Based Scene Reconstruction Using Commodity Graphics Hardware. In *Proceedings of Pacific Graphics 2002*, 225–234.

ZHANG, C., AND CHEN, T. 2004. A Self-Reconfigurable Camera Array. In *Proceedings of Eurographics Symposium on Rendering*, 243–254.

ZHANG, L., CURLESS, B., AND SEITZ, S. 2003. Spacetime stereo: Shape recovery for dynamic scenes. In *Proceedings of Conference on Computer Vision and Pattern Recognition (CVPR)*, 367–374.

ZITNICK, C., KANG, S. B., UYTTENDAELE, M., WINDER, S., AND SZELISKI, R. 2004. High-Quality Video View Interpolation using a Layered Representation. *ACM Transactions on Graphics 23*, 3, 600–608.