Fisheye Video Correction

Jin Wei, Chen-Feng Li, Shi-Min Hu, Ralph R Martin, and Chiew-Lan Tai

Abstract—Various types of video can be captured with fisheye lenses; their wide field of view is particularly suited to surveillance video. However, fisheye lenses introduce distortion, and this changes as objects in the scene move, making fisheye video difficult to interpret. Current still fisheye image correction methods are either limited to small angles of view, or are strongly content-dependent, and therefore unsuitable for processing video streams. We present an efficient and robust scheme for fisheye video correction, which minimizes time-varying distortion and preserves salient content in a coherent manner. Our optimization process is controlled by user annotation, and takes into account a wide set of measures addressing different aspects of natural scene appearance. Each is represented as a quadratic term in an energy minimization problem, leading to a closed form solution via a sparse linear system. We illustrate our method with a range of examples, demonstrating coherent natural-looking video output. The visual quality of individual frames is comparable to those produced by state-of-the-art methods for fisheye still photograph correction.

Index Terms—fisheye lens, wide-angle lens, video correction, distortion, mesh, optimization, least squares minimization

1 Introduction

Like the human eye, fisheye lenses can capture a wide field of view, albeit at the cost of significant barrel distortion. This results in noticeable bending of straight edges, and unnatural appearances of e.g. human faces. While *still* images captured with fisheye lenses are usually created for artistic effect (e.g. shooting broad landscapes to suggest the curve of the Earth), fisheye *videos* are widely used in surveillance monitoring and machine vision [1]. The distinctive curvilinear perspective generated by fisheye lenses may be desirable in fisheye photographs [2]. However, in fisheye videos, the resulting distortion renders them hard to understand and uncomfortable to watch, especially as the distortion changes over time. Recovering *natural looking* results from fisheye videos not only enhances viewing, but also facilitates automated video understanding, since video analysis methods are typically designed to work on distortion-free video.

An intuitive approach to fisheye video correction is to find a global flattening from the view sphere to the view plane, which can then be applied to all frames. However, this is clearly an inadequate solution since the sphere is not a developable surface, so the approach can only work approximately at best, and such problems are exacerbated by the fact that the camera may move during video capture. Computer-aided distortion correction for fisheye (or wide-angle) photographs has been studied by both the computer graphics and computer vision communities. In graphics, several geometric approaches have been proposed, but they are either limited to small angles of view, or are strongly content-dependent, limiting their usefulness in video. In computer vision, various camera calibration methods have been developed for fisheye / wide-angle lenses, but they are only applicable when the field of view is relatively small (typically less than $100°$).

Distortion in fisheye video varies with location within each frame, causing dynamic distortion of moving objects. Attempting to apply existing fisheye photograph correction methods in a frame-by-frame manner leads to unsatisfactory results, either leaving a certain amount of distortion in each frame, or resulting in a lack of temporal continuity in the output. To achieve satisfactory results when processing fisheye video, interframe relationships must be explicitly considered to recover a *natural-looking* result. Moreover, it must do so for time-varying video scenes with widely differing content, without causing any flickering, pulsing or flexing artifacts. Ideally, the correction algorithm should be efficient and perform in real time, allowing non-specialist users to interactively check their results.

Based on Carroll et al.'s recent work on fisheye photograph correction [3], we present an efficient and robust fisheye video correction scheme, using per frame transformations which minimize the time-varying distortions while providing continuity of objects in the video. Each video frame is discretized using a spherical mesh, and its geometric features are represented using a set of anchor points defined in the context of the whole video stream. To obtain natural looking results, six distinct but related correction criteria are used to remove the distortion. They respectively address straight line distortion, boundary consistency, orientation consistency, local shape similarity (i.e. conformality), homogeneity, and temporal coherence. They are expressed in terms of the discretization mesh and the anchor points. These

- *J. Wei and S.M. Hu are with the Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China.*
- *C.F. Li is with the College of Engineering, Swansea University, Swansea SA2 8PP, UK.*
- *R.R. Martin is with the School of Computer Science & Informatics, Cardiff University, 5 The Parade, Roath, Cardiff, CF24 3AA, UK.*
- *C.L. Tai is with the Department of Computer Science and Engineering, the Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong.*

Fig. 1. Swing video. Significant distortions exist in fisheye video (top row). Using a minimization-based technique with an efficient closed-form solution, distortion is effectively removed in a way that coherently preserves content from frame to frame (bottom row). (The horizontal field of view was 160°, and 10 line constraints were used for correction.)

correction criteria are weighted and composed to give a quadratic energy function whose minimization determines the correction mapping, represented as a modification of the discretization mesh. The solution of the resulting least squares minimization problem is obtained in a closed form by solving a sparse linear system. The input video frames are corrected in a streaming manner by warping the discretization mesh frame by frame, the temporal coherence term ensuring temporal continuity.

We use interactive annotation to guide this process. User requirements are expressed as anchor points in a few key frames, and video feature tracking algorithms are used to automatically propagate these user-specified constraints throughout the rest of the video sequence.

Our method outputs smooth, coherent, natural-looking video. We experimentally demonstrate that our method produces results of comparable quality to existing fisheye *image* correction algorithms—while also ensuring temporal coherence. Our method is clearly superior to a frame by frame approach, or the naive method of global viewing sphere flattening. Interactive performance is achieved on a commodity computer.

2 Related Work

Visual information seen from a single view point is defined on a viewing sphere centered at that view point. Thus, correcting fisheye distortion requires finding a correction mapping from the viewing sphere to the image plane. Cartographers have long recognized that mapping a sphere to a plane is difficult: a non-developable surface cannot be flattened onto a planar domain without deformation. They have developed hundreds of projections that trade off different types of distortion, among which are orthographic, stereographic and Mercator projections [4]. However, all of them inevitably include a certain degree of distortion; none of them can satisfactorily recover natural looking images from photos taken with a fisheye camera [3].

Artists have developed specialized techniques for handling large fields of view [2], [5]. Unlike the mappings of cartographers, which are all global and independent of content, artists use multiple local projections to produce paintings with a large field of view [5], [6]. Specifically, the background is often painted from one view point while each foreground person or object is depicted using an additional view point to retain its natural shape.

Based on this multi-projection principle developed by artists, several geometric methods have been developed for correcting fisheye (or wide-angle) still images. Zorin and Barr [7] gave an axisymmetric correction mapping which compromises between bending lines and distorting spheres. Their results are impressive, but their approach is limited to fields of view which can be covered by linear perspective projection. Zelnik-Manor et al. [8] presented a simple method in which a fisheye photo is partitioned and projected onto folding screens. This approach generates sharp discontinuities between projection planes, and so only works for photos which can be readily partitioned along natural discontinuities (e.g. intersections of walls). Kopf et al. [9] extended this method by locally projecting salient regions onto user specified planar polygons, but doing so requires reconstruction of the irregularly deformed cylindrical projection surface in order to prevent discontinuity. Thus, this method cannot be applied in video processing where the image content varies significantly from frame to frame. Recently, Carroll et al. [3] proposed a mesh-based non-linear optimization technique based on minimizing line distortion and preserving local shapes. Their approach can produce visually flawless correction results for many fisheye photographs. However, it cannot be easily adapted to correct frames of a video, because both the correction mapping and the image boundary depend

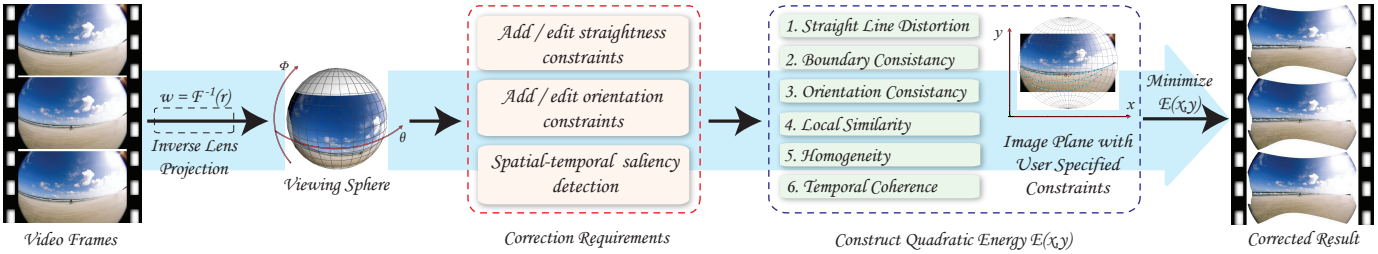


Fig. 2. Framework for fisheye video correction.

strongly on the image content and the user-specified constraints. Furthermore, although a linear approximation is used for speed, it is still too slow to support interactive video processing: a standard PC will take 1–4 hours to process a 10s video clip.

Distortion correction has also been studied in the context of camera calibration [10], [11], [12], [13]. The mapping from the viewing sphere to the image plane is described by internal camera parameters. To account for design and manufacture imperfections, various distortion coefficients are also included in the camera model as internal parameters. The goal of camera calibration is to determine these camera parameters, which can then be used to reconstruct 3D information from 2D images. Direct use of the distortion coefficients for typical lenses allows correction of images to follow the ideal pinhole model. However, for wide-angle and fisheye images, these calibration approaches do not produce pleasant results in that the corrected images inevitably contain severe perspective distortion.

Our fisheye video correction approach builds upon existing mesh parameterization and optimization techniques. In this context, we note that many specific mesh-based optimization techniques have been developed for a range of applications, including image and video retargeting [14], [15] and detail-preserving shape manipulation and deformation [16], [17], [18], [19].

3 Problem and Solution Strategy

With both eyes open, most people can see almost $180°$. Even with this extremely wide angle of view, our eyes perceive a view in which objects retain their natural shapes, so for example, lamp posts appear straight. Fisheye lenses also produce a wide field of view, but with a significant amount of barrel distortion. This distinctive imaging feature is determined by the optical design of fisheye lenses, which we now briefly recap.

A lens can be described by a mapping function from the viewing sphere to the image plane:

$$r = F(\omega), \tag{1}$$

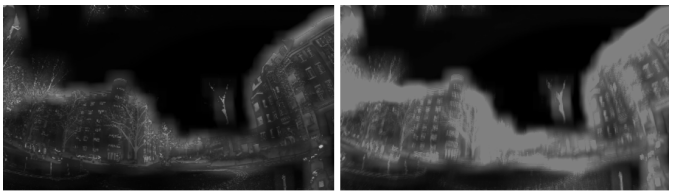
where ω denotes the angle between a projection ray and the optical axis, and r the distance between the projected point and the image center. Nearly all lenses used in photography and video are rectilinear lenses, producing rectilinear (non-spatially-distorted) images of distant objects. Rectilinear lenses with focal length f use a perspective mapping $r = f\tan(\omega)$, and capture images from a small portion of the viewing sphere centered on the optical axis. For angles of view greater than $90°$, rectilinear lenses often suffer from severe perspective distortion [20]. Fisheye lenses are designed to overcome this limitation of rectilinear lenses and achieve a wider field of view, of about $180°$. Many different mapping functions are used in the design of fisheye lenses. Among them, the most popular fisheye projections include equidistant projection $r = f\omega$, equisolid angle projection $r = 2f\sin(\omega/2)$, orthographic projection $r = f\sin(\omega)$ and stereographic projection $r = 2f\tan(\omega/2)$. Due to the fundamental difference between the positive curvature of the viewing sphere, which is thus non-developable, and the zero curvature of the image plane, images created by fisheye lenses have a characteristic convex appearance [1].

We assume we are given a fisheye video stream without knowledge of the camera position, or the geometry of the original 3D scene. However, we do assume that the type and parameters of the fisheye projection performed by the lens are known (this information is provided by the lens manufacturer). The goal of fisheye video correction is to recover a natural-looking video stream with the same content. In general, it is extremely difficult, if not impossible, to reconstruct the viewed 3D scene from arbitrary fisheye videos, so this problem cannot be simply solved by using knowledge of the lens projection. An overview of our solution strategy is now given.

As shown in Fig. 2, our correction scheme reads in consecutive fisheye video frames, and the original visual information on the viewing sphere is readily recovered by an inverse fisheye projection, $\omega = F^{-1}(r)$, using the known lens projection. Fisheye video correction now involves finding a sequence of correction mappings from the viewing sphere onto the image plane. A minimization problem using a specified set of correction criteria reflecting *natural appearance* of the output video stream determines these sphere to plane mappings frame-by-frame to produce a natural-looking video stream.

Six correction criteria are used; all are expressed via quadratic energy functions. The global minimizer of a weighted sum of these energies gives the correction mapping; a closed-form solution can be found by solving a linear system corresponding to a sparse positive-definite

(a) *Straight features* (b) *Constraint propagation*



(c) *Spatial saliency* (d) *Spatio-temporal saliency*

Fig. 3. Continuous correction requirements.

matrix. These correction criteria, taking into account such issues as line distortion and temporal coherence, provide a comprehensive set of measures for different aspects of *natural appearance*, based on both large scale features and local pixel distributions. For ease of representation of the distortion correction, the viewing sphere is divided along lines of longitude and latitude to form a rectangular mesh, allowing the correction mapping to be described by a flexible mesh transformation from the viewing sphere to the image plane.

Some of our correction criteria are designed to preserve salient objects in the video, which requires information concerning the key structures and content of the input video. Experience in the related application of video retargeting shows that fully automated feature detection often fails to retain user-preferred parts [21]. Therefore, we provide an interactive tool for the user to indicate straightness and orientation constraints, and to mark important regions in key frames. These constraints are then propagated into the remaining video frames. This keeps user interaction low, and ensures the necessary temporal coherence in the output.

4 Correction Requirements

It has long been known by artists that global perspective projection cannot be used when painting large fields of view as it causes severe perspective distortion. Instead, artists use different view points for different objects in a large field of view, and have developed various techniques to seamlessly combine multiple local projections in a single painting [2], [5], [6]. This is consistent with the observation that fisheye distortion correction is a data-dependent problem. Furthermore, in fisheye *video* correction, the correction mapping must suit the content of the whole video stream, not just specific individual frames. In order to obtain satisfactory results for a given fisheye video stream, the correction algorithm must be informed about the key structures and content whose natural appearance is to be preserved; we will refer to these as the *correction requirements*. To achieve temporal coherence, the correction requirements should vary in a continuous manner over time (except, of course, at discrete scene changes). These correction requirements are specified in three forms: straight features, orientation requirements, and an importance map.

4.1 Straight Features

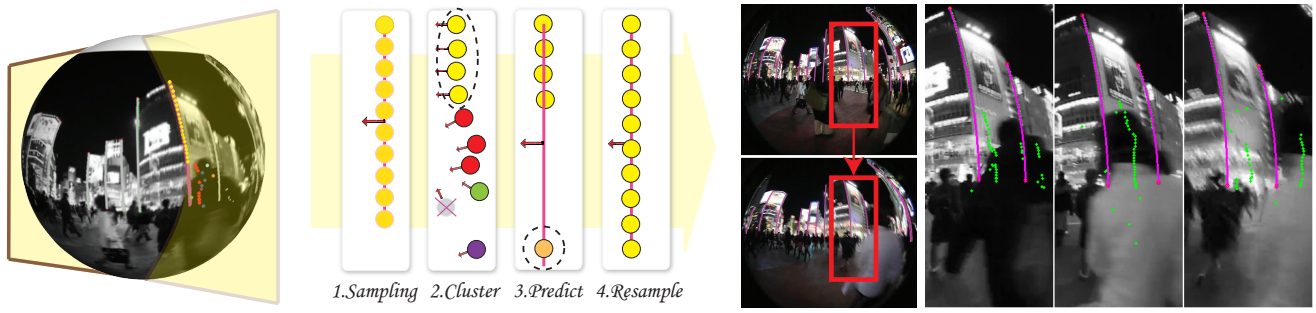
As human eyes are extremely sensitive to straight lines, straight features such vertical corners of buildings should be straightened by fisheye video correction. Such features are first specified by the user in a few key frames. Note that the distorted straight features in the fisheye image are represented as arcs of great circles on the viewing sphere (Fig. 4(a)). Therefore, knowing the fisheye lens projection, straight features in the distorted image can be readily defined by specifying their endpoints. We then propagate the user specified straight lines into other frames to construct a set of continuous straight-line constraints.

Although a straight feature is fully determined by its endpoints, tracking these two endpoints alone is unreliable, in particular when the straight feature is temporarily occluded by another object in the fisheye video. To improve robustness of propagation of this constraint between adjacent video frames, we instead track the whole great circle arc defined on the viewing sphere, and then use information about the line's geometry to extract or predict its two endpoints depending on the occlusion status. Tracking of arcs and extraction of endpoints are cross-checked respectively by local gradient estimation and content registration of fisheye frames, providing a sophisticated mechanism for filtering out errors arising in automatic video tracking.

User-specified straight-line constraints are often not exactly placed on the straight features, but close to them. In fact, distorted lines attracting viewers' attention are normally high-gradient edges. We use this to apply a fine-tuning operation to straight-line constraints before the tracking process. A user-specified straight line is first sampled as a set of points as shown in Fig. 4(a), where the sampling density is proportional to the arc length on the viewing sphere. These points are then moved within a small neighborhood onto the pixels with the highest gradient values, found using the Sobel operator after bilateral filtering. We then map the new points onto the viewing sphere to obtain the best fitting great circle arc, satisfying

$$\arg\min_{\mathbf{n}} = \sum_{s} (\mathbf{q}_s \cdot \mathbf{n})^2, \tag{2}$$

where $\mathbf{n}$ is the normal vector to the plane of the 3D, and $\mathbf{q}_s = (x_s, y_s, z_s)$ is the Cartesian coordinate of the sample point on the viewing sphere. The refined sample points $\mathbf{p}_s$ (including the two endpoints) are then obtained by projecting $\mathbf{q}_s$ onto the fitted arc.



(a) *Defining line constraints* (b) *Propagating line constraints* (c) *Handling temporary occlusions*

Fig. 4. Constraint propagation. A distorted straight line in the fisheye image determines a unique great circle plane on the viewing sphere, as shown in (a). This property is borrowed to reduce tracking errors. A brief workflow for straight-line tracking is shown in (b). An example in which all straight features suffer from multiple occlusions throughout the entire timeline is shown in (c). Zoomed-in results (frames 20, 30, 35) demonstrate the effectiveness of our propagation strategy (green points: Bouget's results; purple lines: predicted straight features).

After refinement of the user input, the projected points $\mathbf{p}_s$ are tracked to the next frame. Video feature tracking is a challenging fundamental problem in computer vision, and has been extensively studied in the last two decades [22]. We use a pyramid-KLT tracker [23]. For general video feature tracking, it is stable and effective with good tracking quality, but failures can arise, especially when there are occlusions of straight features. As obtaining correct straight-line constraints is of key importance to ensure a visually pleasing result, it is essential to use a robust scheme to handle tracking failures. To examine the tracking quality, we first check whether the tracked points still form a unique great circle arc on the viewing sphere (step 2 in Fig. 4(b)). Let $\mathbf{p}_s^{'}$ denote the s-th tracked point of a line. The local normal vector of the arc-plane at this point is computed from the two adjacent vertices on the viewing-sphere as $(\mathbf{p}_{s+1}^{'} - \mathbf{p}_s^{'}) \times (\mathbf{p}_s^{'} - \mathbf{p}_{s-1}^{'})$ (for the endpoints we use the two nearest vertices). Then, similar local normals (within an angular distance of η) are grouped into clusters, and the largest cluster is considered to be correct, and is the *trusted cluster*. Points grouped in all other clusters are treated as errors and abandoned. In the example in step 2 in Fig. 4(b), four groups are generated by the clustering process (the gray point is filtered out before clustering as it exceeds an user-specified error tolerance). The yellow points form the trusted cluster (marked by a dashed ellipse), while the other three groups (red, green and purple) are discarded. The normal vector of the new great circle arc is computed by averaging all local normals as $\mathbf{n}_{arc} = \sum_1^N k_b \mathbf{n}_b / \|\mathbf{n}_b\|$, where N is the number of points in the trusted cluster, and the weight k_b indicates the confidence in point b, determined by comparing the color distance between two patches centered at the corresponding point before and after tracking. For simplicity we set $k_b = 1/N$ for all examples. The average normal vector $\mathbf{n}_{arc}$ indicates a great circle arc on the viewing sphere, corresponding to the new position of the straight feature, which is further improved through local gradient estimation and great circle arc fitting as described in the previous paragraph.

Our experiments show that endpoints can be successfully tracked in most cases because they are usually located at corners with strong contrast. In cases when an endpoint is abandoned during arc tracking, the algorithm will predict a new position for the missing endpoint using line geometry (step 3 in Fig. 4(c)). Specifically, the lost endpoint is predicted to lie along the arc using the angular distance ϑ to the most reliable tracking point $\mathbf{p}_{mb}$, selected from the trusted cluster to be close to the corresponding endpoint and to have lowest change from the previous frame. The change considered here includes the angular distance from its previous position and the average scaling ς between $\mathbf{p}_{mb}$ and its neighbors on the arc. The new position for the missing endpoint is then predicted by rotating $\mathbf{p}_{mb}$ through an angle $\varsigma\vartheta$:

$$\mathbf{p}_{predict} = \mathrm{Rotate}(\mathbf{p}_{mb}, \varsigma\vartheta). \tag{3}$$

Our implementation also handles temporary occlusions caused by moving objects in a fisheye video. The occurrence of occlusion is identified when tracking fails simultaneously for an endpoint and its neighbors. Note that occlusions appearing in the interior of straight features do not affect the construction of new straight-line constraints, and so do not require special treatment. When predicted endpoints are successfully tracked again over two consecutive frames, the period of occlusion is considered to be over. Then, within a small neighborhood of the predicted endpoint and between the last successfully tracked frame and the current frame, deformable feature matching [24] is performed to identify the real endpoint, which is sequentially interpolated with the predicted endpoints in the last few frames in order to obtain a smooth constraint flow. Before applying [24], Mercator projection is used on the two patches to map the spherical patch onto the plane in a conformal manner. Fig. 4(c) shows an example in which all straight features need prediction for the lower

endpoint of a line through the entire timeline shown. The feature lines suffer from constant occlusions from the very start. A series of zoomed-in views show the predictions for the most severely affected area in the video, in which the green points indicate directly tracked sample points using Bouget's approach [23] while the purple lines indicate the predicted lines. This example shows that our strategy is capable of greatly reducing tracking even when severe occlusion occurs.

Special attention is required for straight features that go beyond the frame border, e.g. a flagpole extending upwards out of the frame. We first test if an endpoint has reached the frame border using a threshold d. If the border has been reached, the corresponding endpoint will be set on the border along the computed great circle arc. We also handle the case when the the end of the straight object moves into the frame, e.g. the top of a flagpole moves inside the frame border. Specifically, a small patch near the frame border is compared with the patch centered at the the sample points a few steps ago, and if the difference is greater than a user-specified threshold, the algorithm will report a new endpoint entering the frame and mark it for a new straight-line constraint. When comparing patches, Gaussian weighting is used to enhance the importance of central pixels.

During a fisheye video, some straight lines may move out of the frame, and new straight features may move in. We test in each tracking step if the number of straight-line sample points has reduced by more than 35%. If so, the current frame is set as a key frame, and user interaction is required to mark some new straight-line constraints. The algorithm then propagates these new constraints both forward and backward to improve their smoothness. Our constraint propagation strategy works well for most cases, but in some extreme cases where a large occlusion occurs causing tracking to fail for almost all points, manual input is still needed. Other more intelligent methods are needed to handle such complicated scenarios. As intelligent feature tracking is not the main topic of this paper, we leave this for future consideration.

4.2 Orientation

The orientations of certain important content items should remain constant throughout the video sequence. The user places orientation constraints on chosen dominant straight features such as edges of tall buildings and the horizon. These features are constrained to not only stay straight but also to lie in the prescribed direction. Using linear interpolation, orientation constraints are also propagated through the video stream.

4.3 Importance Map

Mapping images from the fisheye input to the output video inevitably introduces deformation. In order to create natural looking images, it is desirable to concentrate the mapping deformation in less noticeable areas, to allow important features with high visual impact to have the desired shape. Therefore, it is necessary to identify salient content in the input video stream for the correction algorithm. Recent image and video resizing systems [14], [15] have used various saliency detection algorithms. We considered several saliency computation methods [25], [26], [27] for our fisheye video correction scheme, but none of them were suitable for all examples tested. Furthermore, their computational efficiency is often too low for video processing.

Instead, we propose a simple and very efficient strategy for saliency computation, which takes into account spatial importance, temporal coherence and the influence of object movements. First, for every pixel, we calculate the standard deviation in color for a small window around it, to get an importance map for spatial saliency (see e.g. Fig. 3(c)). Secondly, in order to produce coherent output, we also take time variation of saliency into account. The spatial saliency in a k frame window is used (in practice $k = 5$ works well in most cases), following a similar rule to that in [21]. This is designed to squeeze the deformation far away from salient objects through the timeline. Finally, to prevent jittering due to motion of salient objects, we detect motion saliency by estimating LK optical-flow [28]: motion saliency is defined by pixel shifts in the estimated motion field and normalized to $[0, 1]$ over each pair of adjacent frames. To ensure good spatial coherence, the normalized motion saliency of each pixel is reevaluated in a similar way to spatial saliency, and the lower motion saliency is truncated from the final importance computation. Thus, the total importance map value is defined as:

$$W_i^* = \omega_s \sum_{t=i}^{i+k} W_t / (t - i + 1) + \omega_m W_i m_i + 1, \qquad (4)$$

where W_i^* is the total saliency value of a pixel in the i-th frame, W_i its spatial saliency, and m_i its motion saliency. Spatial saliency in the k frame window is weighted according to the frame number, so that the effect of future frames on the current frame reduces over time. Weights ω_s and ω_m are applied to spatial and motion saliency; we set $\omega_s = \omega_m = 2$ in our experiments. An example of the spatio-temporal saliency map W_i^* is given in Fig. 3(d). The total saliency of a mesh vertex is then assigned by calculating the average of W_i^* within a patch around the vertex, and scaled to lie in $[0, 1]$.

We have also investigated saliency due to pixels *moving differently* from surrounding pixels. However, the influence of this type of saliency is relatively small and sometimes even harmful depending on the video clip. Therefore, differences of pixel movements are not considered in Eqn. (4).

5 Correction Criteria

Performing fisheye video correction largely depends on quantifying different aspects of *natural visual appearance*, a very subjective concept. We use six measures, each

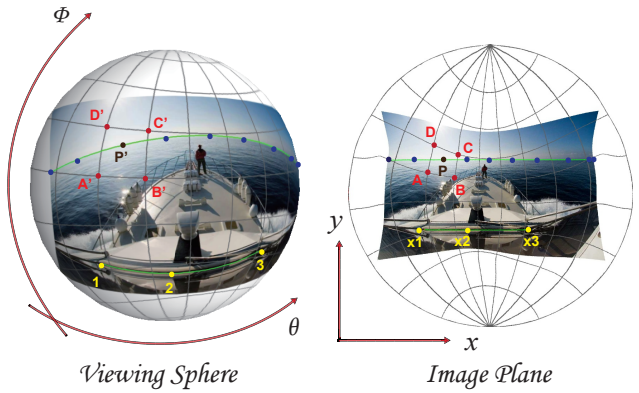


Fig. 5. Parameterization of the correction mapping (for computation, a discretization mesh of 1° angular increments is used, but a sub-mesh of 20° angular increments is shown to illustrate the transformation.)

defined as a quadratic form, to do this. Correction of fisheye videos is posed as finding a mapping which minimizes an overall quadratic objective function. Some criteria depend strongly on the video content, while other criteria are determined by differential relations of the imaging process.

The viewing sphere is parameterized by $(-\pi/2 \leq \theta, \phi \leq \pi/2)$—see Fig. 5. Let (x, y) be Cartesian coordinates in the image plane. We wish to find the correction mapping $x = x(\theta, \phi)$, $y = y(\theta, \phi)$. The viewing sphere is meshed into rectangular cells at equal increments of θ and ϕ with grid coordinates (θ_j, ϕ_i); the correction mapping is correspondingly discretized on the image plane with Cartesian nodal coordinates $x_{i,j}$, $y_{i,j}$. This mapping is determined using a quadratic objective function based on the six correction criteria.

5.1 Straight Line Preservation

Straight features should become straight after fisheye video correction. Such a constraint has long been used in camera calibration, and is typically expressed in terms of squared residual distances of points from their corresponding best-fit lines [10], [11]. However, this results in a non-linear objective function, needing an iterative solver [3], and it is hard to use in video correction which has high requirements of robustness and efficiency. It is also noted that along a straight line, aspect ratios of background objects should remain proportional to their real dimensions in the 3D world. Based on this observation, we precalculate the scaling coefficients along a straight feature using geometric information on the viewing sphere, which in turn allows the straight-line distortion to be measured by a quadratic energy term.

Straight features appear curved in the fisheye image, corresponding to arcs of great circles on the viewing sphere (see Fig. 5). On the viewing sphere, let 1, 2 and 3 denote three sample points on a straight feature specified by the user. In the image plane, let (x_1, y_1), (x_2, y_2) and (x_3, y_3) denote the corrected projections for these three points. These are collinear in the image plane if and only if they satisfy the relations

$$\frac{x_1 - x_2}{x_2 - x_3} = \frac{y_1 - y_2}{y_2 - y_3} = \frac{d_{12}}{d_{23}}, \tag{5}$$

where d_{ij} is the distance between (x_i, y_i) and (x_j, y_j) in the image plane. As these three points lie on a great circle on the viewing sphere, the length ratio d_{12}/d_{23} in Eqn. (5) can be approximately determined as an angle ratio, i.e. $d_{12}/d_{23} \approx \gamma_{12}/\gamma_{23}$ where γ_{ij} is the angle subtended by points i and j on the viewing sphere. This approximation assumes that the length ratio on the image plane can be approximated by the angle ratio on the viewing sphere, which holds particularly well within small angles of view.

The distortion of a straight feature can be simply measured using the sum of squared residuals of the linear relations defined above. Let l denote a user specified straight feature in the fisheye image. Through inverse fisheye projection, the straight feature is mapped onto an arc of a great circle on the viewing sphere, which is then discretized using M_l sample points. The line-distortion energy corresponding to l is defined as

$$E_l = \sum_{m=2}^{M_l - 1} \left\| \gamma^l_{mM_l} \left(\mathbf{x}^l_1 - \mathbf{x}^l_m \right) - \gamma^l_{1m} \left(\mathbf{x}^l_m - \mathbf{x}^l_{M_l} \right) \right\|^2, \tag{6}$$

where $\mathbf{x}^l_m = (x^l_m, y^l_m)$ are Cartesian coordinates of the corrected sample points on the image plane, and γ^l_{mn} is the central angle between sample points m and n on the viewing sphere.

This line-distortion energy function E_l is quadratic in the variables (x^l_m, y^l_m), the Cartesian coordinates of the projected sample points. However, the basic unknowns in the correction mapping are $(x_{i,j}, y_{i,j})$, the nodal coordinates of the discretization mesh on the image plane. To express sample points in terms of the discretization mesh, we use bilinear interpolation [29]: a sample point P located in a quadrilateral cell $ABCD$ in the image plane, has Cartesian coordinates

$$\mathbf{x}_P = \bar{\alpha}\bar{\beta}\mathbf{x}_A + \alpha\bar{\beta}\mathbf{x}_B + \alpha\beta\mathbf{x}_C + \bar{\alpha}\beta\mathbf{x}_D \tag{7}$$

where relative coordinates (α, β) are determined from the corresponding rectangular cell $A'B'C'D'$ on the viewing sphere, and $\bar{\alpha} = 1 - \alpha$, $\bar{\beta} = 1 - \beta$. Substituting Eqn. (7) into Eqn. (6), the energy function E_l becomes a quadratic form in the Cartesian coordinates of the discretization mesh.

5.2 Boundary Consistency

In a normal video stream, all frames share the same image boundary. Hence, we define a new quadratic energy function to fix the image boundary for the output video. Let V_B denote the complete set of points sampled on the boundary of the fisheye image and, for each point $P \in V_B$, let $\mathbf{x}_P = (x_P, y_P)$ denote its Cartesian

coordinates projected on the image plane. The boundary consistency relation is

$$\mathbf{x}_P = \overline{\mathbf{x}_P}, \qquad P \in V_B, \tag{8}$$

where $\overline{\mathbf{x}_P} = (\overline{x_P}, \overline{y_P})$ is the fixed location for P. The energy corresponding to the above boundary consistency relation is defined as

$$E_B = \sum_{P \in V_B} \left(w_P^B\right)^2 \|\mathbf{x}_P - \overline{\mathbf{x}_P}\|^2, \tag{9}$$

where w_P^B $(P \in V_B)$ are fixed weights controlling the rigidity of the image boundary. The boundary energy function E_B is a quadratic form in the Cartesian coordinates of the sample points. After substituting Eqn. (7) into Eqn. (9), E_B becomes a quadratic form in the Cartesian coordinates of the discretization mesh.

5.3 Orientation Consistency

Consistency of selected orientations are next considered. Orientation requirements are specified on dominant straight features, e.g. the horizon should be horizontal in the output. Such constraints can be described by a sequence of sample points located on each prescribed straight line. Let M_o denote the number of sample points on a straight orientation feature o in the fisheye image. The orientation consistency relation is

$$\mathbf{a}_o \cdot \mathbf{x}_P + c_o = 0, \qquad P = 1, \cdots, M_o, \tag{10}$$

where $\mathbf{a}_o = (a_o, b_o)$ is a normal to the user-specified target orientation, c_o an unknown parameter fixing the location of the straight line feature, and $\mathbf{x}_P$ $(P = 1, \cdots, M_o)$ are Cartesian coordinates of the sample points along it. The orientation energy associated with the above consistency relation is defined as

$$E_o = \sum_{P=1}^{M_o} \|\mathbf{a}_o \cdot \mathbf{x}_P + c_o\|^2, \tag{11}$$

which is quadratic in the $\mathbf{x}_P$ and c_o. Again, bilinear interpolation Eqn. (7), is used to express E_o as a quadratic form in the nodal coordinates of the discretization mesh and the unknown c_o.

5.4 Local Similarity

Significant barrel distortion exists in fisheye images due to the optical design of fisheye lenses, so, e.g., square window frames can appear almost elliptical. Such local geometric features should have their natural shapes in the corrected image. Thus, the correction mapping must preserve local similarities from the viewing sphere to the image plane. We use an energy which has previously been used for fisheye photograph correction [3].

On the viewing sphere, an infinitesimal line element with constant ϕ has length $ds_\theta = \cos\phi\, d\theta$, while a line element with constant θ has length $ds_\phi = d\phi$. In the image plane, line elements in x- and y- directions have lengths $ds_x = dx$, $ds_y = dy$ respectively. The Jacobian $\mathbf{h}$ of the mapping gives the local length scaling in each direction upon transformation. To ensure similarity of local geometric features during transformation, line-element scaling must be the same in all directions, so

$$\frac{\partial x}{\partial \phi} = -\frac{\partial y}{\partial \theta}\frac{1}{\cos\phi}, \quad \frac{\partial y}{\partial \phi} = \frac{\partial x}{\partial \theta}\frac{1}{\cos\phi}; \tag{12}$$

these are the Cauchy-Riemann equations for conformal mapping of a sphere to a plane [30].

In terms of the discretization mesh, the above condition for local similarity can be approximated by the finite-difference form:

$$\begin{aligned} x_{i+1,j} - x_{i,j} &= -(y_{i,j+1} - y_{i,j})/\cos\phi_{i,j}, \\ y_{i+1,j} - y_{i,j} &= (x_{i,j+1} - x_{i,j})/\cos\phi_{i,j}. \end{aligned} \tag{13}$$

The squared residuals of the above relations are used to define a quadratic energy function measuring local similarity:

$$\begin{aligned} E_S = &\sum \left(w_{ij}^S\right)^2 [\cos\phi_{i,j}\,(x_{i+1,j} - x_{i,j}) + (y_{i,j+1} - y_{i,j})]^2 \\ &+ \sum \left(w_{ij}^S\right)^2 [\cos\phi_{i,j}\,(y_{i+1,j} - y_{i,j}) - (x_{i,j+1} - x_{i,j})]^2 \end{aligned},$$

where sums are taken over the discretization mesh, and the weights w_{ij}^S adjust the stiffness of local similarities across the fisheye video frame.

5.5 Homogeneity

During correction of fisheye videos, the local similarity criterion above essentially allows local geometric features on the viewing sphere to undergo a rigid motion and an *isotropic* scale change when transformed to the the image plane. To produce a natural looking result, we must avoid large differences in scale change throughout the image, particularly between neighboring features.

Local scaling from the viewing sphere to the image plane is given by the Jacobian $\mathbf{h}$ defined earlier. To obtain *homogeneous* scaling, the gradient of $\mathbf{h}$, $\nabla\mathbf{h}$, must be set to zero, where

$$\nabla\mathbf{h} = \begin{pmatrix} \frac{\partial^2 x}{\partial\theta^2}\frac{1}{\cos\phi} & \frac{\partial^2 x}{\partial\phi\partial\theta}\frac{1}{\cos\phi} + \frac{\partial x}{\partial\theta}\frac{\sin\phi}{\cos^2\phi} & \frac{\partial^2 x}{\partial\theta\partial\phi} & \frac{\partial^2 x}{\partial\phi^2} \\ \frac{\partial^2 y}{\partial\theta^2}\frac{1}{\cos\phi} & \frac{\partial^2 y}{\partial\phi\partial\theta}\frac{1}{\cos\phi} + \frac{\partial y}{\partial\theta}\frac{\sin\phi}{\cos^2\phi} & \frac{\partial^2 y}{\partial\theta\partial\phi} & \frac{\partial^2 y}{\partial\phi^2} \end{pmatrix}.$$

This differential condition for homogeneous local deformations can again be approximated in terms of the discretization mesh, leading to a finite-difference form

$$\mathbf{H}_{i,j} = \begin{pmatrix} (x_{i,j+1} - 2x_{i,j} + x_{i,j-1})\cos\phi_{i,j} \\ (x_{i+1,j+1} - x_{i+1,j} - x_{i,j+1} + x_{i,j})\cos\phi_{i,j} \\ \quad + (x_{i,j+1} - x_{i,j})\sin\phi_{i,j} \\ (y_{i,j+1} - 2y_{i,j} + y_{i,j-1})\cos\phi_{i,j} \\ (y_{i+1,j+1} - y_{i+1,j} - y_{i,j+1} + y_{i,j})\cos\phi_{i,j} \\ \quad + (y_{i,j+1} - y_{i,j})\sin\phi_{i,j} \\ (x_{i+1,j+1} - x_{i,j+1} - x_{i+1,j} + x_{i,j})\cos^2\phi_{i,j} \\ (x_{i+1,j} - 2x_{i,j} + x_{i-1,j})\cos^2\phi_{i,j} \\ (y_{i+1,j+1} - y_{i,j+1} - y_{i+1,j} + y_{i,j})\cos^2\phi_{i,j} \\ (y_{i+1,j} - 2y_{i,j} + y_{i-1,j})\cos^2\phi_{i,j} \end{pmatrix},$$

and in turn, a homogeneity energy:

$$E_H = \sum \left(w_{ij}^H\right)^2 \|\mathbf{H}_{i,j}\|^2, \tag{14}$$

where the weights w_{ij}^H adjust the homogeneity stiffness across the image.

A similar concept (termed *smoothness* by the authors) was introduced in [3] for fisheye *photograph* correction, but a different mathematical formulation was used in an empirical manner. The main advantage of our new homogeneity energy is its robustness and wide applicability to different video frames, as demonstrated in our results later.

5.6 Temporal Coherence

Temporal coherence is an essential requirement in video processing. In this case, it requires the correction mapping for a fisheye video stream to be a smooth function of time. Our correction mapping is a sequence of mesh transformations from the viewing sphere to the image plane, one for each video frame. We must thus ensure that interframe changes in this mapping are sufficiently small to be invisible when viewing the corrected video.

The mesh on the viewing sphere remains fixed, and only the projected mesh on the image plane changes. In successive frames, grid cells on the image plane have different shapes but identical connectivity. The amount of interframe mesh deformation can be expressed by the total *strain energy* of the grid cells [31].

We thus define the temporal coherence energy as

$$E_T = \sum_{e \in V_E} \frac{1}{2} (\mathbf{X}_e^* - \mathbf{X}_e)^{\mathrm{T}} \mathbf{K}_e (\mathbf{X}_e^* - \mathbf{X}_e), \qquad (15)$$

where V_E is the set of quadrilateral grid cells in the mesh, $\mathbf{X}_e^*$ is a vector representing the eight nodal coordinates of cell e for the *current* frame, $\mathbf{X}_e$ represents its nodal coordinates for the *previous* frame, and $\mathbf{K}_e$ is the 8×8 stiffness matrix of cell e determined from the *previous* frame. If matrix $\mathbf{K}_e$ degenerates to an identity matrix, the energy E_T becomes an elementary deformation measure defined as the the sum of squared nodal displacements. Eqn. (15) is a standard finite element definition of strain energy [32]. For completeness, the stiffness matrix $\mathbf{K}_e$ is given without proof: $\mathbf{K}_e = A_e \mathbf{B}_e^{\mathrm{T}} \mathbf{D}_e \mathbf{B}_e$ where A_e is the area of cell e determined from the *previous* frame, $\mathbf{D}_e$ is a 3×3 constant matrix

$$\mathbf{D}_e = \left(w_e^T\right)^2 \begin{pmatrix} 2 & 1 & 0 \\ 1 & 2 & 0 \\ 0 & 0 & 1/2 \end{pmatrix}, \qquad (16)$$

where the weight w_e^T adjusts the temporal-coherence stiffness of cell e, and $\mathbf{B}_e$ is a 3×8 constant matrix

$$\mathbf{B}_e = \begin{pmatrix} u_1 & 0 & u_2 & 0 & u_3 & 0 & u_4 & 0 \\ 0 & v_1 & 0 & v_2 & 0 & v_3 & 0 & v_4 \\ v_1 & u_1 & v_2 & u_2 & v_3 & u_3 & v_4 & u_4 \end{pmatrix}. \qquad (17)$$

The above matrix entries are defined as

$$\begin{pmatrix} u_1 & u_2 & u_3 & u_4 \\ v_1 & v_2 & v_3 & v_4 \end{pmatrix} = \frac{1}{4} \mathbf{J}_e^{-1} \begin{pmatrix} 1 & -1 & -1 & 1 \\ 1 & 1 & -1 & -1 \end{pmatrix},$$

where $\mathbf{J}_e$ is the 2×2 matrix

$$\mathbf{J}_e = \begin{pmatrix} 1/4 & -1/4 & -1/4 & 1/4 \\ 1/4 & 1/4 & -1/4 & -1/4 \end{pmatrix} \begin{pmatrix} x_1^e & y_1^e \\ x_2^e & y_2^e \\ x_3^e & y_3^e \\ x_4^e & y_4^e \end{pmatrix},$$

determined by the nodal coordinates $(x_i^e, y_i^e)\ \ i = 1, \cdots, 4$ of cell e in the *previous* frame.

This temporal coherence energy is a quadratic form in the nodal displacements of the projection mesh for the *current* frame. It can be intuitively viewed as a weighted sum of squared nodal displacements, where the weights appear in the form of the constant stiffness matrix $\mathbf{K}_e$ which is determined by the mesh configuration in the *previous* frame. Comparing to a direct sum of nodal displacements (i.e. setting $\mathbf{K}_e = \mathbf{I}$), this definition is more precise: the simpler measure cannot distinguish between different deformations caused by grid nodes moving along different directions but by the same distance. In our limited experiments, we have found that the Mahalanobis distance in Eqn. (15) consistently performs better than an unweighted temporal term (i.e. $\mathbf{K}_e = \mathbf{I}$).

For the first frame, there is no previous frame, so we simply set $E_T = 0$.

5.7 Minimizing the Total Energy

Six quadratic energy functions have been defined above, measuring different aspects of *natural appearance* of the output video stream. We combine these to give the overall objective function for fisheye video correction:

$$E = \sum_{l \in V_L} \left(w_l^L\right)^2 E_l + E_B + \sum_{o \in V_O} \left(w_o^O\right)^2 E_o + E_S + E_H + E_T,$$

where V_L denotes the complete set of straight features, V_O denotes the complete set of orientation features, the weight w_l^L controls the impact of straight feature l, and the weight w_o^O controls the impact of orientation feature o. This is a quadratic form in the nodal coordinates of the discretization mesh and the location parameters c_o in the orientation energy E_o. The correction mapping represented by the discretization mesh on the image plane is given by the global minimizer of E.

All correction criteria represented by the six energy terms above must be carefully balanced using appropriate weight factors (which have fixed values throughout the video). The straight line factor w_l^L, the boundary factor w_P^B and the orientation factor w_o^O mainly affect the global appearance of the video, while the local similarity factor w_{ij}^S, the homogeneity factor w_{ij}^H and the temporal coherence factor w_e^T have more local effects. As we aim to preserve straightness, the straight line and orientation terms should be dominant in the total energy, and we typically choose these weights as 100. Local similarity and homogeneity control dispersion of distortion throughout the image, and although we could allow user control, in practice it is desirable to set these proportional to the importance map W_i^* (see Eqn. (4)),

multiplied by 1 and 10 respectively. To achieve a smooth result over time, we find it is best to set the temporal coherence weight to a fixed value of 15 in most cases. The boundary weight is less important as it only affects the overall shape of the corrected image, and we simply set it to 1.

The overall quadratic objective function depends only on the mesh transformations of a small window of frames for each successive video frame, allowing our approach to correct fisheye video in a streaming manner, frame by frame (with reinitialization at scene changes). Flickering and related artifacts commonly encountered in video processing are avoided through three measures. First, the user specified correction requirements are enforced on each frame, keeping shapes smooth. Secondly, future saliency is taken into account by estimating the temporal context using a window of $k+1$ frames; the resulting latency is unimportant. Thirdly, the addition of the temporal coherence energy term allows the best visual quality to be achieved as a balanced consensus.

The quadratic objective function E can be minimized using a symmetric positive-definite linear system (i.e. $\mathbf{A}\mathbf{x} = \mathbf{b} \neq \mathbf{0}$) by setting its derivatives to zero. As each node is only directly linked to its neighboring nodes in the formulation of E, a *sparse* linear system results. The boundary condition in Eqn. (8) ensures *uniqueness* of the solution. The major computational cost in our approach to fisheye video correction is in solving this linear system; standard linear solvers are used. Having a closed-form solution, our least squares formulation is extremely efficient: it avoids issues of slow convergence, instability and local minima that can be encountered in some nonlinear approaches using iterative solvers.

6 Results and Discussion

We have tested our system with many examples, confirming the effectiveness of our method for transforming fisheye videos into natural looking output streams. Video clips taken with different types of fisheye lenses were used in testing, with horizontal angles of view ranging from 160° to 180°, covering half of the entire viewing sphere or just the central part of it. All input videos contained severe barrel distortion, leading to visibly misshapen human faces and bent straight lines. For single still images, we have verified that our algorithm produces results comparable to state-of-the-art fisheye *image* correction methods, while for video correction, the new method is clearly superior to a naive extension of existing image correction methods, in terms of both visual quality and performance.

6.1 Image Correction Comparison

First, we compare our approach to still fisheye *image* correction methods. Intuitively, one might expect that corrected results should have the shape expected under perspective projection, in which straight lines appear naturally straight. However, the human visual system tends to concentrate on the central part of the scene, and people rarely notice the fact that perspective projection actually leads to extreme stretching when the visual angle is large. Conformal spherical projections often trade off different types of distortion, and are suitable when it is desirable to preserve small objects (e.g. human faces) in a large scene, so we also make a comparison with Mercator and stereographic projections.

Fig. 6 shows an example comparing our results with those of various other still fisheye distortion correction methods. Global projections do not give a satisfactory effect. In conformal projections, small faces are preserved well, but it is clear that neither projection completely removes distortion. Mercator projection corrects vertical lines well, but fails on horizontal lines lying across the visual field. Stereographic projection trades off different types of distortion, and works best for scene lines crossing the optical axis of the lens. Carroll's content-preserving projection is perhaps the best for still fisheye photo correction, but the results appear stretched near the poles, distorting objects nearby. This artifact is partially caused by the lack of direction protection for image boundaries, and partially by the lack of control of line scaling during straight line correction. (For a fair comparison we chose the same line marks as Carroll.) Our result combines the advantages of these methods, in general. By introducing a boundary energy term into the quadratic objective function, the boundary shape is protected by enforcing it to be a circle, as in stereographic projection. Along straight features, the aspect ratios of background objects are retained by using geometric information on the viewing sphere to determine the line scaling, but the distorted lines and compressed poses are corrected.

6.2 Video Examples

When performing fisheye video correction, we must eliminate distortion while keeping fixed scene elements (like background buildings) stable. It is clear that global projection does not consider this problem. Our supplemental video shows that our method is able to produce coherent and visually pleasant video output streams. A number of examples of correcting fisheye video frames in different situations are provided, including a video with a fixed background and moving objects, and others with moving cameras and both fixed and moving content. We compare our results with perspective projection, Zorin's one parameter correction [7], and a naive extension of Carroll's non-linear optimization image correction approach [3], where the video stream is corrected frame by frame with the same user annotations as those in the new method. The tuning parameters and weights required in [7] and [3] were manually adjusted to give the best results. In addition, for a fair comparison, we excluded specific saliency measurements like face and object detection to allow the comparison to be performed on a common basis.

Fig. 6. Output of various still fisheye photograph correction methods.

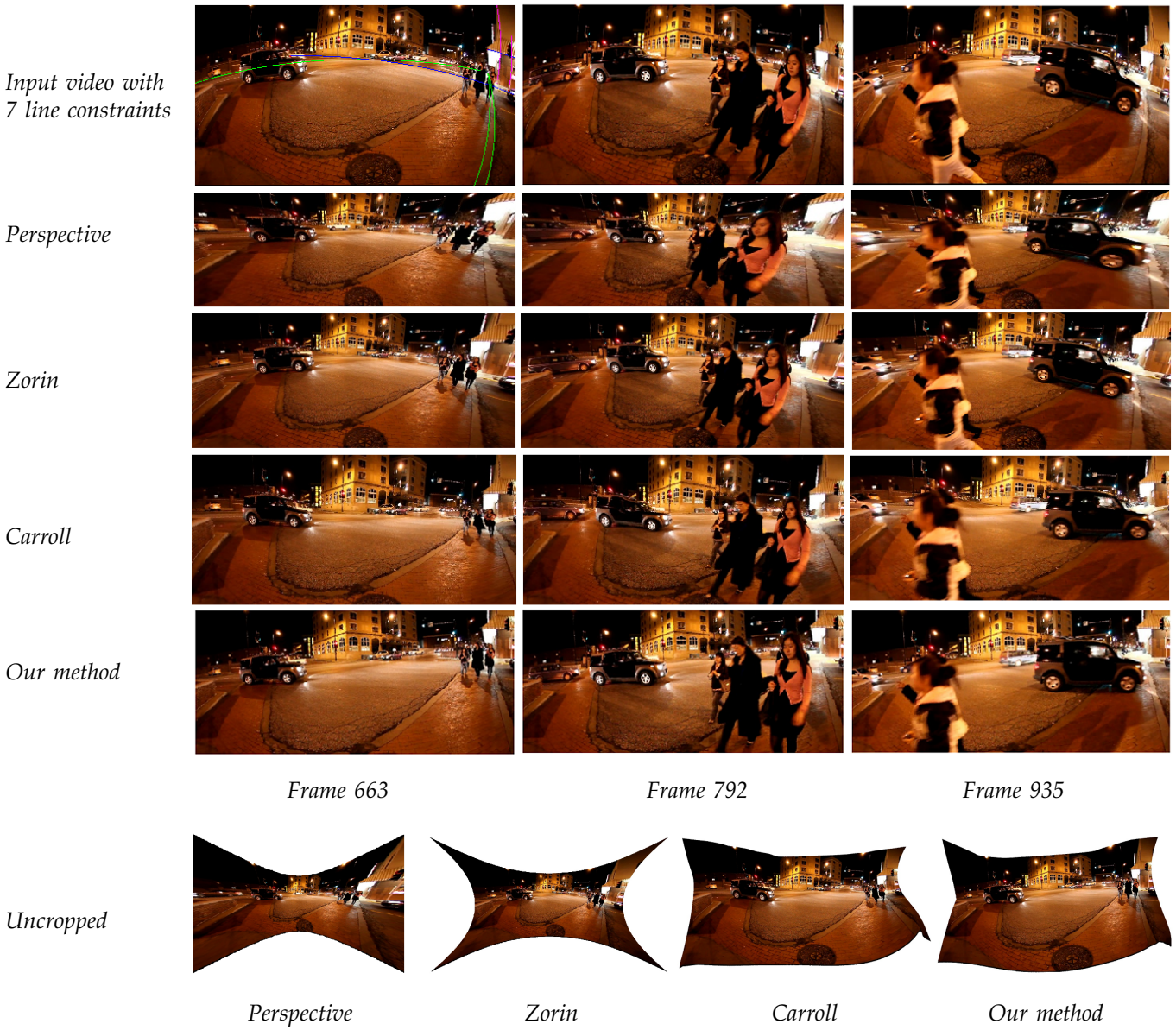
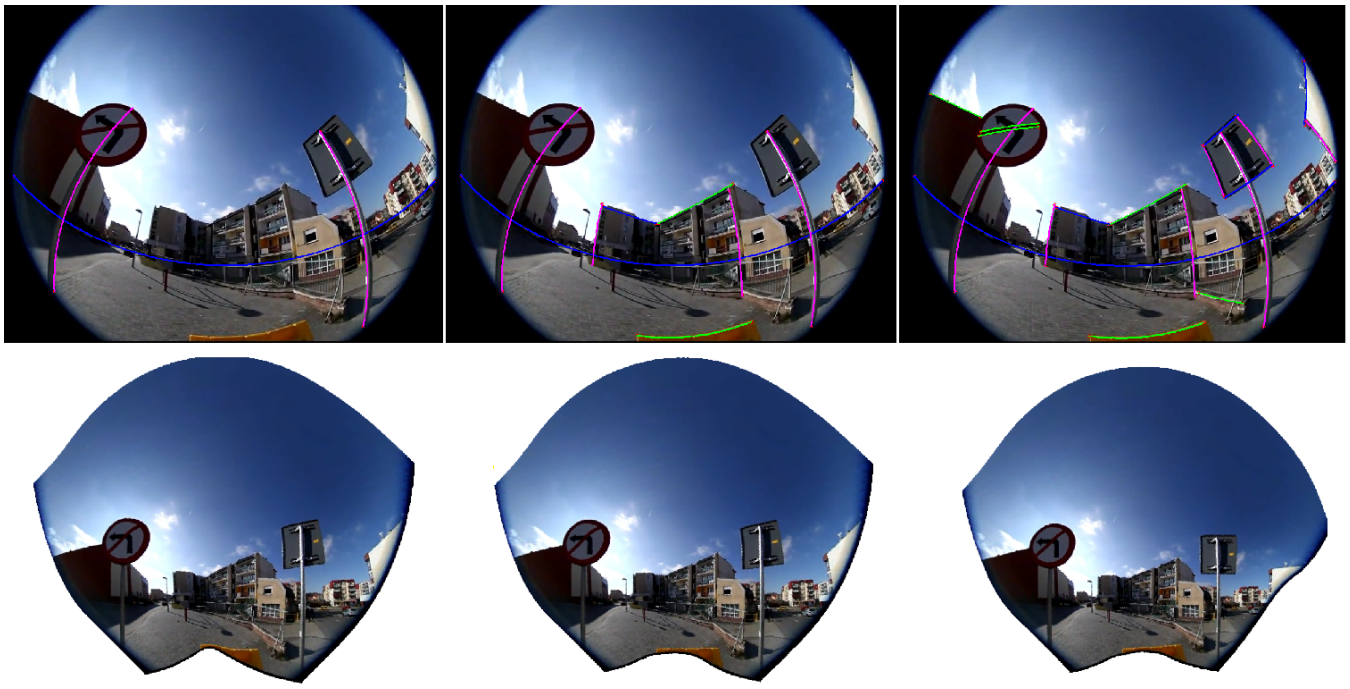


Fig. 7. Output of various fisheye video correction approaches (horizontal angle of view: $160°$).

Fig. 7 shows three video frames generated by each of these methods (also see our accompanying video). For perspective projection, the stretch distortion changes dramatically when the object or camera moves, making the output hard to watch and difficult to follow. Zorin's method trades off different types of cartographic projection, but due to its global nature, distortions still remain. The naive extension of Carroll's method produces visually acceptable results for most individual frames, but when played back as a video stream, severe flickering

Metro, 16 line constraints *Without orientation constraints* *With orientation constraints*

Fig. 8. Effectiveness of orientation constraints (horizontal angle of view: $160°$).



3 lines *8 lines* *18 lines*

Fig. 9. An example of interactive fisheye video correction. Global visual quality is determined by three crucial orientation constraints, while details are refined by adding further minor constraints (horizontal angle of view: $180°$).

artifacts occur due to lack of spatio-temporal coherence. In the naive extension of Carroll's method, objects distort when many straight constraints are close to each other, a phenomenon which does not occur in our approach. Further results in the supplemental video demonstrate the effectiveness of our spatio-temporal coherence energy terms.

The example in Fig. 8 (also see the supplemental video) further shows the flexibility of our algorithm in allowing control over the overall effects in the corrected video, by specifying fixed orientations for lines. This helps to produce stable results as perceived by human vision. This is particularly helpful when the horizon is a major feature in the scene.

6.3 Performance and Usability

We tested our algorithm on a PC with an Intel 2.5Ghz Core 2 Duo CPU and 2GB memory. The algorithm speed depends mainly on the mesh density. In all cases shown in this paper, fisheye video frames were mapped onto the front half of the viewing sphere, divided into a spherical mesh with 181 by 181 (about 32700) vertices. To assess the performance of our method, it was compared with the naive extension of Carroll's method [3], in which frame optimizations are solved using a linearized two-step iteration scheme. On this PC platform, our approach supports interactive video processing, with a runtime of around 0.42s per frame. The naive extension of Carroll's method can sometimes produce visually comparable results for individual frames after 2–3 iterations, but takes on average 8–10 two-step iterations for the optimization procedure to converge; a single iteration step usually costs more than 1s, giving an average time per frame of over 12s. The computational cost of Carroll's method varies linearly with the number of iterations, while our new approach is faster and takes a constant time (due to its closed form solution). As video correction algorithms

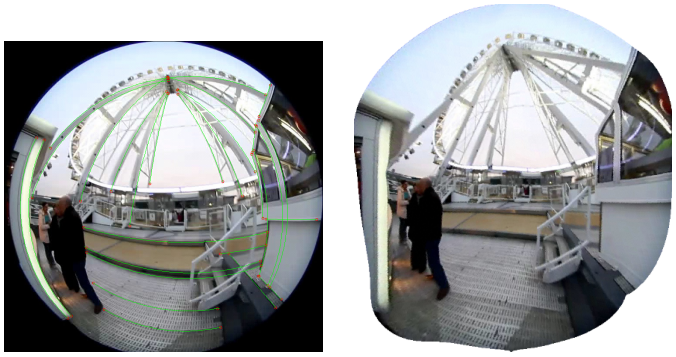


Fig. 10. An unsuccessful case in which a large circular structure remains distorted; note that several straight line constraints pass through the circle.

must work with frames having significantly different contents and constraints, a fixed and predictable high performance is a desirable property, especially in an interactive scenario. We believe that our method could provide real-time performance using GPU acceleration, but this is beyond the scope of the present work and is left for future investigation.

Our fisheye video correction system is easy to use for novice users. As shown in Fig. 9, global visual quality is determined by a few dominant straight-line and orientation constraints, and the influence of minor line constraints is more local yet still intuitive. In our investigation, new users (graduate students in computer science without a particular background in image editing) were almost immediately able to produce acceptable results by following a few simple rules to ensure that dominant line and orientation constraints do not conflict. After a simple introduction from the authors and a short self-training session (20–30 minutes), novice users were able to obtain good results for many new video clips using only a few trial runs.

6.4 Limitations

Our method has some minor limitations which we now discuss. As we correct distortion by warping a spherical mesh with user specified constraints, a sufficiently large mesh must be used to allow user constraints to be satisfied. Typically 40 constraints can be met with a 181 by 181 vertex mesh. Increasing the mesh resolution will provide more freedom for distortion correction and consequently allow more line constraints in the image, but a larger linear system results, requiring an increased solution time. Once the mesh resolution is sufficient for correcting the distortion marked by line constraints, further increases in mesh density provide little extra benefit in terms of visual quality.

We explicitly try to prevent global straight line distortion in the output (i.e. to ensure that straight lines are straight), but there is no guarantee that distortion will not noticeably affect other simple shapes covering large areas of the scene (see e.g. the funfair wheel in Fig. 10). Our algorithm also relies on successful video feature tracking. Current video tracking algorithms tend to fail when processing video with significant discontinuities caused by occlusion, and objects leaving or entering the scene, so manual adjustments are inevitably needed.

While boundary consistency is ensured in the corrected video stream, the frame boundaries often appear as irregular shapes as a result of ensuring the best visual quality for salient video content (see e.g. Fig. 7). We could additionally use video completion methods to present a rectangular video, but for simplicity we just let users crop a rectangular region manually.

Our correction algorithm also assumes knowledge of the lens projection (or that it can be estimated by fitting a pre-existing camera model).

7 Conclusions and Future Work

We have presented a novel approach to constructing optimized transformations which turn distorted fisheye video into a natural-looking sequence. Our method provides an effective and efficient approach to eliminate distortion from a continuous video clip. Our method preserves salient objects and straight features in a spatiatemporally coherent manner, leading to natural looking output video which is easy to watch.

The energy terms defined in our system could also be used in other applications such as image resizing and camera calibration. A possible extension of our work is to perform explicit foreground-background segmentation, to separate salient objects from possibly highly detailed but unimportant background. Finally, a more sophisticated saliency detection strategy, and automatic line detection, could improve the overall visual effects while minimizing the user interaction required.

Acknowledgment

We would like to thank the anonymous reviewers for their constructive comments. This work was supported by the National Basic Research Project of China (No. 2011CB302205) and the Natural Science Foundation of China (No. U0735001. The authors would also like to thank the EPSRC, and the Royal Society and Natural Science Foundation of China for support through an International Joint Project (No. JP100987).

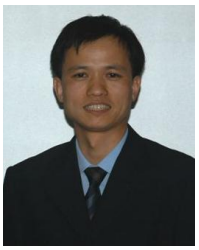
References

[1] J. P. Hornak, *Encyclopedia of Imaging Science & Technology*. Wiley-Interscience, 2002.
[2] A. Flocon and A. Barre, *Curvilinear Perspective from Visual Space to the Constructed Image*. University of California Press, 1988.
[3] R. Carroll, M. Agrawala, and A. Agarwala, "Optimizing content-preserving projections for wide-angle images," *ACM Trans. Graph.*, vol. 28, no. 3, p. 43, 2009.
[4] J. P. Snyder, *Flattening the Earth: Two Thousand Years of Map Projections*. University Of Chicago Press, 1997.
[5] M. Kubovy, *The Psychology of Perspective and Renaissance Art*. Cambridge University Press, 1988.
[6] M. H. Pirenne, *Optics, Painting & Photography*. Cambridge University Press, 1970.

[7] D. Zorin and A. H. Barr, "Correction of geometric perceptual distortions in pictures," in *SIGGRAPH '95*, 1995, pp. 257–264.
[8] L. Zelnik-Manor, G. Peters, and P. Perona, "Squaring the circle in panoramas," in *ICCV '05*, 2005, pp. 1292–1299.
[9] J. Kopf, D. Lischinski, O. Deussen, D. Cohen-Or, and M. Cohen, "Locally Adapted Projections to Reduce Panorama Distortions," *Comput. Graph. Forum*, vol. 28, no. 4, pp. 1083–1089, 2009.
[10] R. Swaminathan and S. Nayar, "Nonmetric calibration of wide-angle lenses and polycameras," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 10, pp. 1172–1178, 2000.
[11] F. Devernay and O. Faugeras, "Straight lines have to be straight - Automatic calibration and removal of distortion from scenes of structured environments," *Machine Vision and Applications*, vol. 13, no. 1, pp. 14–24, 2001.
[12] J. Kannala and S. S. Brandt, "A generic camera model and calibration method for conventional, wide-angle, and fish-eye lenses," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 8, pp. 1335–1340, 2006.
[13] R. Hartley and S. B. Kang, "Parameter-free radial distortion correction with center of distortion estimation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 8, pp. 1309–1321, 2007.
[14] G.-X. Zhang, M.-M. Cheng, S.-M. Hu, and R. R. Martin, "A shape-preserving approach to image resizing," *Comput. Graph. Forum*, vol. 28, no. 7, pp. 1897–1906, 2009.
[15] Y.-S. Wang, H. Fu, O. Sorkine, T.-Y. Lee, and H.-P. Seidel, "Motion-Aware Temporal Coherence for Video Resizing," *ACM Trans. Graph.*, vol. 28, no. 5, p. 127, 2009.
[16] T. Igarashi, T. Moscovich, and J. Hughes, "As-rigid-as-possible shape manipulation," *ACM Trans. Graph.*, vol. 24, no. 3, pp. 1134–1141, 2005.
[17] S. Schaefer, T. McPhail, and J. Warren, "Image deformation using moving least squares," *ACM Trans. Graph.*, vol. 25, no. 3, pp. 533–540, 2006.
[18] J. Huang, X. Shi, X. Liu, K. Zhou, L.-Y. Wei, S.-H. Teng, H. Bao, B. Guo, and H.-Y. Shum, "Subspace gradient domain mesh deformation," *ACM Trans. Graph.*, vol. 25, no. 3, pp. 1126–1134, 2006.
[19] M. Ben-Chen, O. Weber, and C. Gotsman, "Variational harmonic maps for space deformation," *ACM Trans. Graph.*, vol. 28, no. 3, pp. 1–11, 2009.
[20] S. F. Ray, *Applied Photographic Optics*. Focal Press, 2002.
[21] P. Kraehenbuehl, M. Lang, A. Hornung, and M. Gross, "A system for retargeting of streaming video," *ACM Trans. Graph.*, vol. 28, no. 5, p. 126, 2009.
[22] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," *ACM Comput. Surv.*, vol. 38, 2006.
[23] J.-Y. Bouguet, "Pyramidal implementation of the lucas kanade feature tracker description of the algorithm," 2000.
[24] B. Georgescu, S. Member, P. Meer, and S. Member, "Point matching under large image deformations and illumination changes," *IEEE Trans. Pattern Anal. Machine Intell*, vol. 26, pp. 674–688, 2004.
[25] T. Liu, J. Sun, N.-N. Zheng, X. Tang, and H.-Y. Shum, "Learning to detect a salient object," in *in: Proceedings of IEEE Conference on Computer and Vision Pattern Recognition (CVPR*, 2007, pp. 1–8.
[26] C.-L. Guo, Q. Ma, and L.-M. Zhang, "Spatio-temporal Saliency Detection Using Phase Spectrum of Quaternion Fourier Transform," in *in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.
[27] S. Goferman, L. Zelnik-Manor, and A. Tal, "Context-Aware Saliency Detection," in *in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.
[28] B. D. Lucas and T. Kanade, "An Iterative Image Registration Techniue with an Application to Stereo Vision ," in *Proceeding of Imaging Understanding Workshop*, 1981, pp. 121–130.
[29] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes 3rd Edition: The Art of Scientific Computing*, 3rd ed. Cambridge University Press, 2007.
[30] M. Floater and K. Hormann, "Surface parameterization: a tutorial and survey," in *Advances in Multiresolution for Geometric Modelling*, 2005, pp. 157–186.
[31] T. J. R. Hughes, *Mathematical Foundations of Elasticity*. Dover Publications, 1994.
[32] O. C. Zienkiewicz, R. L. Taylor, and J. Z. Zhu, *The Finite Element Method: Its Basis and Fundamentals*, 6th ed. Elsevier Ltd., 2005.

Jin Wei is currently a Research Assistant at the Department of Computer Science and Technology, Tsinghua University. He received his MS degree in Computer Science from Tsinghua University and BS degree from Huazhong University of Science and Technology. His research interests include digital geometry modeling and processing, video processing and computational camera.

Chen-Feng Li is currently a Lecturer at the College of Engineering of Swansea University, UK. He received his BEng and MSc degrees from Tsinghua University, China in 1999 and 2002 respectively and received his Ph.D. degree from Swansea University in 2006. His research interests include computer graphics and numerical analysis.



Shi-Min Hu is currently a Chair Professor in the Department of Computer Science and Technology, Tsinghua University, Beijing. He received the PhD degree from Zhejiang University in 1996. His research interests include digital geometry processing, video processing, rendering, computer animation, and computer-aided geometric design. He is on the editorial board of *Computer Aided Design* and Associate Editor of *Computer & Graphics*. He is a member of the IEEE, ACM and CCF.

Ralph Martin is currently a Professor at Cardiff University. He obtained his PhD degree in 1983 from Cambridge University. He has published more than 200 papers and 12 books, covering such topics as solid and surface modeling, intelligent sketch input, geometric reasoning, reverse engineering, and various aspects of computer graphics. He is a Fellow of: the Learned Society of Wales, the Institute of Mathematics and its Applications, and the British Computer Society. He is on the editorial boards of *Computer Aided Design*, *Computer Aided Geometric Design*, *Geometric Models*, the *International Journal of Shape Modeling*, *CAD and Applications*, and the *International Journal of CADCAM*.

Chiew-Lan Tai is currently an Associate Professor of Computer Science at the Hong Kong University of Science and Technology. Her research interests include geometry modeling and processing and computer graphics. She received her BSc degree in mathematics from the University of Malaya, MSc in computer & information sciences from the National University of Singapore, and DSc degree in information science from the University of Tokyo.