AUTO-RECTIFICATION OF USER PHOTOS

Krishnendu Chaudhury, Stephen DiVerdi, Sergey Ioffe

krish@google.com, diverdi@google.com, sioffe@google.com

ABSTRACT

The image auto rectification project at Google aims to create a pleasanter version of user photos by correcting the small, involuntary camera rotations (roll / pitch/ yaw) that often occur in non-professional photographs. Our system takes the image closer to the fronto-parallel view by performing an affine rectification on the image that restores parallelism of lines that are parallel in the fronto-parallel image view. This partially corrects perspective distortions, but falls short of full metric rectification which also restores angles between lines. On the other hand the 2D homography for our rectification can be computed from only two (as opposed to three) estimated vanishing points, allowing us to fire upon many more images.

A new RANSAC based approach to vanishing point estimation has been developed. The main strength of our vanishing point detector is that it is *line-less*, thereby avoiding the hard, binary (line/no-line) upstream decisions that cause traditional algorithm to ignore much supporting evidence and/or admit noisy evidence for vanishing points.

A robust RANSAC based technique for detecting horizon lines in an image is also proposed for analyzing correctness of the estimated rectification.

We post-multiply our affine rectification homography with a 2D rotation which aligns the closer vanishing point with the image Y axis.

Index Terms— image, vanishing points, affine rectification, homography, projective geometry, RANSAC

1. INTRODUCTION

Few amongst us can take a perfect photo. One relatively common mistake is to introduce small involuntary camera rotations (about an arbitrary axis) leading to perspective and other distortions. The goal of this project is to automatically fix these distortions and take the image closer to the fronto-parallel view (a fronto-parallel view corresponds to a camera orientation such that an upright rectangular object in the scene, e.g., a door or window, is rectangular in the image, [1]). Specifically, we perform an affine rectification ([1]) which restores parallelism of the lines that are parallel in the fronto-parallel view. The 2D homography corresponding to our affine rectification is computed from a pair of estimated vanishing points ([1]). Towards this end, a new *line-less* RANSAC ([2]) based approach to vanishing point estimation has been developed. Avoiding line detection improves the accuracy and robustness of the vanishing point detector. Finally, we post-multiply the above homography with a 2D rotation. This rotation aligns the vanishing point closer to image Y axis (in an angular sense) with the image Y axis. It can be easily proved ([1]) that post-multiplication by rotation does not destroy an affine rectification (i.e., the rotation does not undo the parallel line restoration).

It should be noted, that we do not perform metric rectification which restores angles. Thus, for instance, right angles in the fronto-parallel view that got distorted, may not get restored in our rectified image. Metric rectification requires detection of three vanishing points or knowledge of orthogonal line pairs. But most Google photos do not have enough structure to support detection of 3 vanishing points. Nor do we have knowledge, in general, of any orthogonal lines. Our experiments over a large number of photos indicate that the proposed approach strikes the best balance between viewing quality and hit-ratio.

In this paper, we describe our system, with special emphasis on the vanishing point estimator.

2. RELATED WORK

Photo rectification via manual warping (also known as Keystoning) has been available via image editing softwares (e.g., Adobe Lightroom) for some time. Rectification via special hardware, at the time of image capture, (e.g., tilt-shift lenses [3]), is also available. An automatic method for adjusting in-plane rotations can be found in [4]. Our approach, on the other hand, is fully automatic, does not need special hardware and is not restricted to in-plane rotations only. In [5] a fully automatic method is presented for improving visual quality of photographs containing man-made structures. Here, the authors propose a set of quantitative criteria for visual quality (picture frame alignment, eye-level alignment, perspective distortion and distortion of rectangular objects) and estimate a homography that is optimal with respect to these criteria via an energy minimization framework. We, on the other hand, have adopted a more direct approach of computing two vanishing points and directly estimating the

affine homography. Our approach to rectification is closer in philosophy to [6, 7] except that we do not need knowledge of any angle or length ratio in the image.

In this paper, we also introduce a *line-less* RANSAC vanishing point detection technique. Vanishing point detection is a much researched topic. The first vanishing point detector was developed in 80s ([8]), using Hough Transform and Gaussian Spheres. Its accuracy was highly sensitive to the choice of Hough bins. In [9], an EM (Expectation Maximization) based approach was proposed. But this was sensitive to the initialization of the EM. In [10], a voting based approach is presented. Here, a large number of finite non-zero length line segments are detected. The set of intersection points between all pairs of these line segments are candidate vanishing points and the winner is chosen via a voting scheme. The search space is pruned via criteria like "a vanishing point cannot be internal to an existing line segment". Search for the winning candidate is guided by orthogonality, camera and vanishing line criterion. This technique lends itself to RANSAC easily. Our experiments indicate that the pre-processing step of line detection imposes significant loss of information and at the same time introduces much noise in the system. The vanishing point detector proposed by us uses a voting like [10], but does not need lines. Also, the approach in [10] cannot be easily simplified to detecting two vanishing points only, whereas ours is symmetric enough to detect one, two or three vanishing points as required. Also, unlike [10], we do not need to handle finite and infinite vanishing points separately. Our experiments indicate, overall, this approach produces more visually appealing results on a wider variety of images. The techniques presented in this paper are covered under a pending Google patent application ([11]).

3. OUR APPROACH

Our overall approach has the following stages: (i) Edgelet Computation (ii) Vanishing Point Detection (iii) Computation of Rectification Homography (iv) Feasibility Analysis We will, now, take a deep dive into each.

3.1. Edgelet Computation

An edgelet is an abstract entity, attached to each edge point of the image, with 3 properties: (i) edge location (the image coordinates of the edge point) (ii) edge direction (unit vector along the edge) (iii) edge strength To compute edgelets, we run a Harris corner detector ([12]) style operator on the image. We use a 3 x 3 window to compute derivatives. Eigen values and eigen vectors of the Gaussian weighted covariance matrix corresponding to rectangular 5 x 5 neighborhood around each pixel are computed. Edgelets are extracted from only those points where one eigen value is big and the other small - these are the the pure edge points (as opposed to corners where both eigen values are big and uniform brightness zones where both eigen values are small). Mathematically, an edgelet is represented as $E = \{\vec{x}, \vec{d}, s\}$. where $\vec{x}$ represents the homogenous coordinates[1] for the edge pixel location [1], $\vec{d}$ represents the edge direction in homogenous coordinates (derived from the principal eigen vector of the covariance matrix) and s is the edge strength (principal eigen value of the covariance matrix). An *edgelet line*, $\vec{l}_E$, corresponding to an edgelet E is defined as the line passing through $\vec{x}$ and parallel to $\vec{d}$. The computed edgelets are stored in an edgelets array, descending sorted on edge strength.

3.2. Vanishing Point Estimation

Our vanishing point detector is RANSAC based. Generally speaking, in RANSAC, one hypothesizes a random model and computes consensus for that model by gathering votes for the model. The model with the maximum number of votes wins. This is robust against outliers compared to direct least square regression.

RANSAC Model: In our RANSAC vanishing point detector, a model comprises of a pair of randomly selected edgelets (henceforth referred to as model edgelets). *The hypothesis vanishing point corresponding to the model is the point of intersection of the the edgelet lines corresponding to the two model edgelets*. Thus, given a model $M(E_1, E_2)$ comprising of edgelets $E_1(\vec{x_1}, \vec{d_1}, s_1)$ and $E_2(\vec{x_2}, \vec{d_2}, s_2)$, the hypothesis vanishing point $\vec{v}_M = \vec{l}_{E1} \times \vec{l}_{E2}$ (where $\times$ denotes the vector cross-product) is the intersection point of the edgelet lines $\vec{l}_{E1}$ and $\vec{l}_{E2}$ (in Homogenous mathematics, cross-products of homogenous vectors corresponding to lines yield their intersection point in homogenous coordinates [1]). Note that $\vec{v}_M$ may be at infinity (indicated by a zero third coordinate). The cross product is always computable - except in the degenerate cases, e.g., when the edgelet lines for the model edgelets coincide or the random selector picks the same edgelet twice as model edgelets. As soon as a (random) new model is generated, we check for degeneracy and if degenerate, we reject that model and generate another one. For performance reasons, we do not make the model edgelet selection completely random. Instead, we select the first model edgelet randomly from the top 20 percentile of the edgelets array (remember this array was sorted on edge strength) and the second model edgelet from the top 50 percentile. This effectively biases the system towards stronger edges.

RANSAC Consensus Building (voting): Given model $M(E_1, E_2)$, we iterate over all the other edgelets $E_i(\vec{x_i}, \vec{d_i}, s_i)$. Each E_i casts a vote for the model M

$$vote(E_i, M(E_1, E_2)) = \begin{cases} \frac{1-e^{-\lambda \cos^2 \theta}}{1-e^{-\lambda}} & \text{if } \theta <= 5^\circ; \\ 0 & otherwise. \end{cases}$$

where θ is the smaller angle between the voting edgelet line $\vec{l}_{Ei}$ and the line joining that edgelet's location $\vec{x_i}$ to the hy-

[1] unless specified otherwise, all coordinates and line equations in this paper are specified as Homogenous 2D vectors

pothesis vanishing point $\vec{v}_M$. λ is a system parameter. In other words, the vote is proportional to the direction coherence. It attains its maximum value of 1 when the voting edgelet line passes through the hypothesis vanishing point ($\theta = 0$), dropping to zero as θ reaches 90 degrees. In practise, we clip the vote to zero when this angle exceeds a threshold (5°). The hypothesis/model garnering maximal consensus yields the estimated vanishing point.

Re-estimation of Vanishing Point from best model inliers: Once the consensus gathering phase is over and we have identified the best model, we re-estimate the vanishing point more accurately via a weighted least squares regression. We estimate the optimal (in a least square sense) intersection point for all the inlier edgelet lines corresponding to the best model. Let $S = \{E_i \mid vote(E_i, M_{best}) > 0\}$ denote the set of inlier edgelets corresponding to the best model M_{best}. And, let $\vec{v}^*_M$ denote the (as yet undetermined) optimal vanishing point. Ideally, the edgelet line, $\vec{l}_{Ei}$ will pass through $\vec{v}^*_M$, yielding $\vec{l}_{Ei} \cdot \vec{v}^*_M = 0$ (in homogenous mathematics, zero dot product indicates line-membership of point [1]). If $\vec{l}_{Ei} = \begin{bmatrix} a_i & b_i & c_i \end{bmatrix}^T$, we get $\begin{bmatrix} a_i & b_i & c_i \end{bmatrix} \vec{v}^*_M = 0$. Furthermore, we weight each equation by the vote cast by the corresponding edgelet (strong voters pull the solution closer to themselves). Hence, each inlier edgelet to the best model yields an equation of the form $w_i \begin{bmatrix} a_i & b_i & c_i \end{bmatrix} \begin{bmatrix} x^*_M & y^*_M & z^*_M \end{bmatrix}^T = 0$ where $w_i = vote(E_i, M_{best})$ and $\vec{v}^*_M = \begin{bmatrix} x^*_M & y^*_M & z^*_M \end{bmatrix}^T$. Overall, we end up with the overdetermined homogeneous linear system

$$Diag(w_1, w_2, \cdots w_N) \begin{bmatrix} a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \\ \cdots & & \\ a_N & b_N & c_N \end{bmatrix} \begin{bmatrix} x^*_M \\ y^*_M \\ z^*_M \end{bmatrix} = 0$$

which is homogeneous linear system that we solve for $\vec{v}^*_M = \begin{bmatrix} x^*_M & y^*_M & z^*_M \end{bmatrix}^T$ via the well known *Singular Value Decomposition technique*.

Estimation of the second Vanishing Point: Once the first vanishing point is estimated, we delete all its inlier edgelets from the edgelets array and repeat the process outlined above to estimate the second vanishing point.

3.3. Computation of Rectification Homography

Restoring parallel lines: In $\mathbb{P}^2$, the space of Homogenous 2D points, the *Line at Infinity* stands for the line containing all the points at infinity, i.e., all points of the form $\vec{p}_\infty = \begin{bmatrix} x & y & 0 \end{bmatrix}^T$. Homogenous representation of the *line at infinity* is $\vec{l}_\infty = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T$. It is easy to verify that $\vec{l}_\infty \cdot \vec{p}_\infty = 0$, which proves that all points on infinity lie on the line at infinity.

Geometrically speaking, an image taken with an arbitrarily rotated camera can be thought of as a mapping (2D homography) from the fronto-parallel image plane to some other image plane. It should be noted that here we are talking of camera rotation about any axis, not just the focal axis (thus, potentially, perspective distortion is introduced). We use an affine rectification which makes the lines that were parallel in fronto-parallel view to become parallel again in the image. In other words, our rectification transforms the image back to the fronto-parallel plane, up to an affine transform. The homography corresponding to this rectification is computed as follows ([1]):

Let $\vec{v}_1$ and $\vec{v}_2$ be a pair of vanishing points (estimated, for instance, by the method outlined in 3.2). The line joining them is $\vec{l}_{12} = \vec{v}_1 \times \vec{v}_2$ (in homogenous geometry, cross product yields the join of two points). Then,

$$H = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ l_a & l_b & l_c \end{bmatrix}$$

is our rectification homography where $\vec{l}_{12} = \begin{bmatrix} l_a & l_b & l_c \end{bmatrix}^T$. This homography transforms the vanishing points back to infinity,

$$H\vec{v}_1 = \begin{bmatrix} \cdots \\ \cdots \\ \vec{l}_{12} \cdot \vec{v}_1 \end{bmatrix} = \begin{bmatrix} \cdots \\ \cdots \\ \vec{v}_1 \times \vec{v}_2 \cdot \vec{v}_1 \end{bmatrix} \begin{bmatrix} \cdots \\ \cdots \\ 0 \end{bmatrix}$$

since a scalar triplet product with a repeated vector is identical to 0. Similarly, $H\vec{v}_2$. The curious reader can also verify that the line joining the vanishing points, $\vec{l}_{12}$, too gets transformed back to infinity by this homography (a homography H transforms a line as H^{-T} and the reader can easily verify that $H^{-T}\vec{l}_{12} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T$). Warping the image with H effectively restores parallelism of lines that are parallel in the fronto-parallel view.

Aligning Verticals: We align the near vertical vanishing point with the image Y axis, which, heuristically, often improves visual quality. This is effectively applying a 2D rotation on the image, which does not "undo" the previously done rectification (i.e., restored parallel lines remain so). Let $\vec{v} = \begin{bmatrix} v_x & v_y & 0 \end{bmatrix}^T$ be the post-affine-rectification vanishing point that is closer (in an angular sense) to the image Y axis, $\vec{Y} = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}^T$. Then $\theta = \cos^{-1}\left(\frac{\vec{v}\cdot\vec{Y}}{\|\vec{v}\|\|\vec{Y}\|}\right) = \cos^{--1}\left(\frac{v_y}{\sqrt{v_x^2+v_y^2}}\right)$ is the angle of rotation and the rotation matrix is

$$R = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The overall Homography: The product of the affine rectification and the vertical alignment homographies, i.e., $T = RH$, is used to warp the image. The warped image is cropped and scaled back to match the original image size.

Fig. 1. Image 1.

3.4. Feasibility Analysis

We analyze the result image for possible visual distortions, and bail out of the rectification if necessary. In particular, we do face detection. If, on application of homography T, the aspect ratio of the face bounding box changes beyond threshold, we bail out on that image. Also, we have built a RANSAC and edgelet based *horizon detector*. It hypothesizes a horizon direction by randomly picking up an edgelet with horizontal (within tolerance) direction and examines support for it from other edgelets. If the consensus for the winning hypothesis exceeds a pre-determined threshold, we say the image has a horizon. Thus, essentially, the horizon detector looks for a long line, not necessarily continuous (in fact, constituent segments do not have to be perfectly aligned), whose direction is horizontal (within tolerance). If such a line is detected, we call it horizon. If the angle of an estimated horizon with the X axis *increases* when we apply the rectifying homography, we bail out on that image.

4. PERFORMANCE AND RESULTS

The proposed auto rectification was tried on 2199 random Google+ images (used with owner permissions). On 218 of these, not enough vanishing points were detected, 1233 had bad angle between detected vanishing points, 173 had image size changing too much, 116 had too much rotation, 109 had too much face distortion, 4 had too much horizon distortion. The algorithm successfully delivered rectified image on the remaining 346 images (15.7%).

Fig. 2. Auto Rectified Image 1.

Fig. 3. Image 2.

Fig. 4. Auto Rectified Image 2.

5. REFERENCES

[1] Hartley H. and Zisserman A., *Multiple View Geometry in Computer Vision*, Cambridge University Press, Cambridge, United Kingdom, 2000.

[2] Martin A. Fischler and Robert C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, June 1981.

[3] *Tilt-shift Photography*, wikipaedia.

[4] Gallagher A., "Using vanishing points to correct camera rotation in images," in *Computer and Robot Vision*, 2005, pp. 147–151.

[5] Hyunjoon Lee, Eli Shechtman, Jue Wang, and Seungyong Lee, "Automatic upright adjustment of photographs," in *Proc. CVPR 2012*, 2012, pp. 877–884.

[6] D. Liebowitz and A. Zisserman, "Metric rectification for perspective images of planes," in *IEEE Conference on Computer Vision and Pattern Recognition*, 1998, pp. 482–488.

[7] D. Liebowitz and A. Zisserman, "Combining scene and auto-calibration constraints," in *IEEE International Conference on Computer Vision*, 1999.

[8] Stephen T. Barnard, "Interpreting perspective images," *Artif. Intell.*, vol. 21, no. 4, pp. 435–462, Nov. 1983.

[9] Antone M. E. and Teller S. J., "Automatic recovery of relative camera rotations for urban scenes," in *Proc. CVPR 2000*, 2000, pp. 282–289.

[10] Carsten Rother, "A new approach for vanishing point detection in architectural environments," in *In Proc. 11th British Machine Vision Conference*, 2000, pp. 382–391.

[11] Chaudhury K. and DiVerdi S., "Auto rectification for google+ images," in *Google Patent Application (pending)*, 2013.

[12] Harris C. and Stephens M., "A combined corner and edge detector," in *4th Alvey Vision Conference*, 1988, pp. 147–151.