Pan360: INS Assisted 360-Degree Panorama

Lun-Cheng Chu
HTC Research, Taipei
seamyself@gmail.com

Andre Chen
HTC Research, Taipei
andre.hl.chen@gmail.com

Yu-Hsin Lin
HTC Research, Taipei
peach09142000@gmail.com

Yu-Mei Chen
HTC Research, Taipei
yumei_chen@htc.com

Scott Chien-Hung Liao
HTC Research, Taipei
scottcliao@gmail.com

Edward Y. Chang
HTC Research, Taipei
edward_chang@htc.com

ABSTRACT

We propose a practical stitching method in dealing with input images with weak salient points, high luminance variations, and repetitive structures in an artificial environment. We address these panorama-stitching challenges by inferring orientation of capturing pose by inertial sensors inside a mobile device. Our contributions include identifying an effective way to consider orientation, providing novel methods to further improve feature-matching rate, and solving the vignette effect in low light environments. Experimental results demonstrate that our global alignment method outperforms other stitchers, and the color tone is well maintained and more appealing.

Categories and Subject Descriptors

I.4 [**IMAGE PROCESSING AND COMPUTER VISION**]: Applications

Keywords

Image Stitching

1. INTRODUCTION

Panoramic image stitching has been studied for several decades [15]. Fig. 1 depicts a general stitching pipeline, which is composed of two primary procedures: *global alignment* and *panorama composition*. The *global alignment* procedure assumes that the input images for stitching have the same orbit capture center. Feature descriptors are first extracted from images to find corresponding image features in a pairwise fashion [3]. Then the RANSAC (random sample consensus) algorithm is applied to reduce the outliers in the feature pairs. In the bundle adjustment step, these correspondences are used to estimate each image's camera poses. The *panorama composition* procedure renders all images into one parorama image when all individual poses have been computed. Blending is an important step in composition, since each image may have a different parametric setting.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from Permissions@acm.org.
ImmersiveME'15, October 30, 2015, Brisbane, Australia.
© 2015 ACM. ISBN 978-1-4503-3745-8/15/10 ...$15.00.
DOI: http://dx.doi.org/10.1145/2814347.2814349.

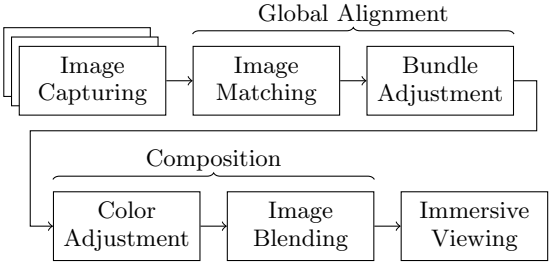


Figure 1: Pipeline of panorama stitching

The color adjustment step is optional since some blending algorithms such as multi-band blending [4] maintains smooth color gradients.

Challenges remain in the above image-stitching pipleline. First, finding correct image-feature correspondences is critical in global alignment. The RANSAC algorithm, however, can fail in several poor-feature conditions such as aliasing and noise. Aliasing is a condition caused by repetitive patterns often appearing in artificial structures (e.g. pavement tiles), which generate congruent features. Noisy features are caused by factors such as low lighing or shaking hands. Second, full view panorama image ($2\pi \times \pi$) usually involves large exposure differences between images. Bands blending is a general approach to alleviate this problem, but the rendering result is not appealing to human perception (e.g., ringing effect around the edge). High dynamic range panorama [6] is a better rendering approach, but the cost in labeling each pixel using graph cut is computationally expensive.

In the work, we present our Pan360 scheme, which takes advantage of the inertial navigation units (accelerometer and gyroscope) equipped with almost all mobile phones to infer the orientation of the camera poses of each frame. Once the camera poses are known, mistakes in feature registration and color adjustment can be reduced effectively and efficiently. Pan360 has been deployed via Google Play and also as an embedded feature of the HTC camera application. Extensive user study demonstrates high quality panorama images produced by Pan360; our online version [1] provides additional sample panorama images for reference.

2. OUR APPROACH

The novelties of Pan360 lie in two aspects. First, we address how inertial sensors help feature selections in the pipeline, and propose novel methods to further improve the image-feature matching accuracy and efficiency. Second, in-

stead of using multi-band blending, we consider the scene radiance and camera parameters in color adjustment. The benefits of our approach are two folds: the blending method can be linear and efficient, and the composition result is appealing to human perception.

Before using sensors to assist frame stitching, sensor readings must be improved by sensor calibration and sensor fusion, which are depicted in Section 2.1. Comparing to the approach proposed by Yang et al. [17], Pan360 performs calibration before every panorama-capturing session. Based on the fusion result, our image matching approach is explained in Section2.2. We show how to recover scene luminance and solve the vignette effect in low light in Section 2.3. Section 2.4 presents our capturing and viewing interface designed for immersive experience on mobile device.

2.1 Sensor Calibration / Fusion

Measured Accelerometer-sensor signal denoted as $\tilde{a}(t)$ and Gyro-sensor signal $\tilde{\omega}(\tau)$ are used to infer camera orientation $\tilde{\theta}(t)$, where $\tau, t \in \mathbb{N}$, the time stamp of a measurement. Assuming the device is held in horizontal, these signals are measured in 3 axes, where y direction is up to the sky, z is out of the screen, and the orientation is counter clockwise. Note that the $\tilde{a}(t)$ and $\tilde{\omega}(\tau)$ may not synchronize.

The relationship between $\tilde{\theta}(t)$ and $\tilde{a}(t)$ is depicted in Eq.(1); the y direction is not available due to its being parallel to gravity.

$$\tilde{\theta}_x(t) = atan(\frac{\tilde{a}_z(t)}{\tilde{a}_y(t)}),\ \tilde{\theta}_z(t) = atan(\frac{-\tilde{a}_x(t)}{\tilde{a}_y(t)}). \quad (1)$$

The relationship between $\tilde{\theta}(t)$ and $\tilde{\omega}(t)$ is depicted in discrete formula Eq.(2), where $\omega(\tau)$ is true signal, ω_b is a constant bias due to non-perfect sensor manufacturing or implantation, and $\delta(\tau)$ is the time lapse between two measurements.

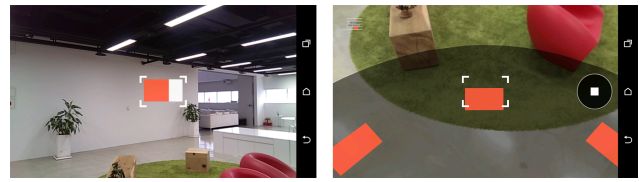
$$\tilde{\theta}(t) = \theta_0 + \sum_{\tau=1}^{t} \tilde{\omega}(\tau)\delta(\tau) = \theta_0 + \sum_{\tau=1}^{t} (\omega(\tau) - \omega_b)\delta(\tau). \quad (2)$$

Calibration: We consider the accumulated error induced by ω_b to be the primary factor that degrades panorama stitching quality. Given a time window $T \in \mathbb{N}$ for gyro measurements in a stationary state, and regarding each measurements as *i.i.d.* samples following Normal distribution $\mathcal{N}(\omega_b, \sigma^2)$, the maximum likelihood estimation on

$$p(\{\omega(\tau)\}_{\tau=1..T} | \omega_b, \sigma^2) = \prod_{\tau=1}^{T} p(\omega(\tau) | \omega_b, \sigma^2)$$

gives $\omega_b = (\sum_{\tau=1}^{T} \omega(\tau))/T$. The UI presented in Fig.2(a) shows that Pan360 requires a user to hold the camera steadily to perform calibration. The retangle in the middle of the screen is filled with orange color from its left to the right in time T, during which gyro measurements are collected and ω_b computed. This simple and timely calibration performed before each panorama capturing sequence remedies the fact that ω_b can change in time and is temperature dependent [17].

Fusion: Calibration ensures accurate sensor readings. Fusing gyro and G-sensor provides Pan360 the estimate of θ_y. Besides, we need rotation response $\omega(t)$ for estimating gravity direction $a(t)$. We use quaternion representation for robust rotation operation. Let $q \in \mathbb{R}^4$, $q_{a(t)} = (0, a_x(t), a_y(t), a_z(t))$, $q_{\omega(\tau)} = (\delta(\tau), \omega_x(\tau), \omega_y(\tau), \omega_z(\tau))$, and $q_{\theta(t)} = (0, \theta_x(t), \theta_y(t), \theta_z(t))$.



(a) (b)

Figure 2: Online Calibration

Fig.2a: The rectangle indicator requires user to hold device stably in time window T for estimating ω_b. Fig.2: Animated targets guides user to aim.

Let θ_g denote the initial device direction (ground direction). As the the device rotates, $\theta_g(t)$ rotates accordingly, and can be calculated in Eq.(2) by setting $\theta_0 = \theta_g$. Then $\tilde{\theta}(t)$ in Eq.(2) can be regarded as $\tilde{\theta}_g(t)$ calculated by the gyro measurements. However, the device direction can also be calculated by acceleration in Eq.(1), and from which the direction is without accumulated error. Hence, the goal of sensor fusion is to remove the accumulated error of the gyro by comparing device direction calculated from G-sensor.

The process of fusion keeps iterating through three steps. In step 1, we assume the accumulated gyro error is neglectable within a few time stamps, and use it to update the direction of acceleration $a(t)$ for quick response. In step 2, we calculate the updated device direction $\tilde{\theta}(t)$ by gyro reading. In step 3, we update the orientation by difference (the cross product) between $\tilde{\theta_g}(t)$ and $\tilde{\theta_a}(t)$, which is the device direction calculated by the acceleration measurement.

1. Update $a(t)$: $q'_{a(t)} = q_{\omega(\tau)} q_{a(t)} q^{-1}_{\omega(\tau)}$;
2. Update $\tilde{\theta}_g(t)$: $\tilde{\theta}_g(t) = \theta_g + \sum_{\tau=1}^{t} \tilde{\omega}(\tau)\delta(\tau)$;
3. Update $\tilde{\theta}$: $q'_{\hat{\theta}} = q_{\tilde{\theta}_g \times \tilde{\theta}_a} q_{\tilde{\theta}(t)} q^{-1}_{\tilde{\theta}_g \times \tilde{\theta}_a}$.

Guiding UI: Fig.2(b) shows how Pan360 instructs a user to take photos one after another by providing sensor-guided targets. Regarding camera position as a center of virtual sphere, the positions of these targets are on the sphere surface. In polar coordinate, let position of target $t_{ij} = (r, \theta_i, \phi_j)$ be the target position. We can express ϕ_j as $\phi_j = \frac{180(j+0.5)}{N+1}, 0 \leq j \leq N$, where $j, N \in \mathbb{N}$. We can then compute $\theta_i = L(\phi_j)$, where L is a lookup table to distribute targets' positions equally in the horizontal direction to achieve the required overlapped *ratio* (our *ratio* setting is 0.5). N and L relate to camera fov (known from smartphone system API) and our desired overlap *ratio* of two adjacent images.

2.2 Image Matching

Algorithm 1 outlines the image matching algorithm of Pan360. The choice of feature operation (line 3) is described in Sec.2.2.1. In Sec.2.2.2 we highlight where the orientation is considered in the algorithm. In Sec.2.2.3 we disclose the methods for further improving the matching rate.

2.2.1 Feature Operation

We use FAST detector and BRIEF descriptor [13] because of their computational efficiency. FAST can be noisy, but it enjoys high true positive rates, which is critical in detecting

Algorithm 1: Image Matching

```
Data: Current Image CurImg
      Current Orientation CurOri
      Previous Data PreData
Result: Image Pair Set PairSet
1  begin
2      PreSet = ∅
3      FeatureOperation(CurImg)
4      for (PreImg, PreOri) ∈ PreData do
5          if Overlap(CurOri, PreOri) OR
6             DeltaTime(CurImg, PreImg) < T then
7              PreSet = PreSet + (PreImg, PreOri)
8          end
9      end
10     ImgPairs = MPLSH(CurImg, CurOri, PreSet)
11     PairSet = ∅
12     for p ∈ ImgPairs do
13         if image pair p pass the RANSAC selection
           then
14             AreaFilter(p)
15             PairSet = PairSet + p
16         end
17     end
18     PreData = PreData + (PreImg, PreOri)
19     return PairSet
20 end
```

features in texture-sparse areas. For descriptor matching, we use Multi-Probe LSH (MPLSH, line 10) [10], which is time efficient (a hash table lookup).

2.2.2 Orientation

We first use the orientation information to prune out all non-neighboring images (line 5). Next, orientation is applied for measuring feature similarity via MPLSH. In our implementation, multiple matches are possible if several descriptors' orientations/positions are congruent. We follow the standard RANSAC procedure. We notice that seldom does RANSAC filter out features, because most potential outliers have been discarded in our descriptor matching step, thanks to the orientation information provided by the sensors.

2.2.3 Gaining Accuracy of Feature Match

First, we allow multiple matches for each feature instead of the best match of MPLSH. Allowing multiple matches increases the chance of finding the correct match, but requires more iterations in RANSAC to remove those wrong ones. Nevertheless, correct match is important to reach globally agreed-upon camera poses in bundle adjustment (Fig.1)

Second, while RANSAC removes feature outliers, it can suffer from false removal of matching features distorted by noise (e.g., features distorted by lense). We apply AreaFilter (line 14) to reduce the number of inliners while maximizing the space coverage. The algorithm of AreaFilter is listed in Alg.2 aiming to have better distribution of points. Too much features cluttering in one region would lead to over-fitting in bundle adjustment step, therefore we reduce the number of feature points while trying to achieve maximum area.

2.3 Color Adjustment

While many methods designed for SLR cameras assume known camera response[2], dense correspondence[8], vignette coherence over all images [7] or a minimum overlapping area

Algorithm 2: AreaFilter

```
Data: Image Pair p
1  begin
2      Set Inliers = p.img1.features
3      Set H = ConvexHull(Inliers)
4      if Area(Hull) < MinArea then
5          return
6      end
7      while Area(Hull) > MinArea/4 do
8          Remove 1 point in H with minimum loss of area.
9      end
10     while Area(Hull) < MinArea do
11         Randomly add 1 point back.
12     end
13     return
14 end
```

[9], these are not available to most handheld devices. We propose replacements of these absent parameters, and the accurate pixel sampling plays an important role here; the assistance of inertial sensors is the key to correctly sample the pixels for color adjustment. The remainder of this subsection describes each component of our color adjustment scheme. We use luminace of image in YUV to avoid imbalanced chrominance artifacts[16]. We solve the exposure in two steps: locally and globally; each image is assigned a local exposure from bundle adjustment, then a global exposure is applied to change the overall luminace.

2.3.1 Image Formation

The relationship between luminance y corresponding to scene point X and irradiance E_X:

$$y = f\left(kE_X\right), \tag{3}$$

where k is the exposure value, which encodes gain value, exposed time and aperture value. Function f is the camera response function. Considering the vignnete effect, rewrite the irradiance E as the product of scene radiance L and vignette correction function M:

$$y = f\left(kM\left(\theta\right)L_X\right), \tag{4}$$

where $\theta = atan(x_p/f)$, x_p is the distance between the pixel's position and image center, and f is focal length (in pixel unit). M is the vignette model [14]: $M(\theta) = 1+v\cdot(cos^4(\theta)-1)$, where v is the vignette parameter to be estimated.

2.3.2 Exposure and Vignette Estimation

To solve the exposure and vignette parameters, consider the fact that two luminance values in image pair $\{i, j\}$ from the same scene point X has the same radiance value [5]:

$$\frac{f^{-1}\left(y_{X_i}\right)}{k_i M_i} = \frac{f^{-1}\left(y_{X_j}\right)}{k_j M_j}, \tag{5}$$

where y_X denotes the luminance y corresponding to X. The advantage of this method is that we can avoid solving radiance L rather than solving radiance, exposure and vignette paramters altogether and thus yeilds to long iterations and estimation steps as in [7], which is time-consuming (average 5 seconds in their test cases with most 11 images while our method computes 40-image cases within 2 seconds).

Let y^j_{k1} denote the kth illuminance value on the first image and y^j_{k2} the kth illuminance value on the second image of jth

image pair. We define

$$\tau = f(\frac{f^{-1}\left(y_{k2}^{j}\right) k_1^j M_1^j}{k_2^j M_2^j}), \quad \tau' = f(\frac{f^{-1}\left(y_{k1}^{j}\right) k_2^j M_2^j}{k_1^j M_1^j}). \quad (6)$$

Then, formulate the error function as $err = \sum_{j=1}^{m} \sum_{k=1}^{n} |\tau - y_{k1}^j| + |\tau' - y_{k2}^j|$, where m is the number of image pairs and n is the number of sample pairs and solve it by Levenberg-Marquardt method.

2.3.3 Global Exposure Correction

Equation (6) shows that it is not possible to uniquely solve for the exposure parameters when the radiance is unknown. Therefore, the global exposure correction must be applied to the result after exposure and vignette estimation. Our global exposure correction refers to Zone System proposed by Ansel Adam, which has a precise evaluation of scenes tones [11]. Rewrite (3) and (4): $y^i = f\left(k^i E_X\right) = f\left(k^i k^g M\left(\theta\right) L_X\right)$, where k^i is the local exposure value for image i and k^g is the global exposure value. To find the global exposure value k^g, the first step is to find the mid tone of the radiance from the whole scene, then we force higher tones to fall in Zone VI and Zone VII [11]. through function z:

$$z(x) = \begin{cases} 205 & \text{if } x > 205 \\ 172 & \text{if } 172 < x \leq 205 \\ x & \text{if } x \leq 172. \end{cases} \quad (7)$$

We compute mean of luminace $\bar{y}$ from all images, $\bar{y} = \frac{\sum_{i=1}^{t} \bar{y^i}}{t}$, where $\bar{y^i}$ is mean luminace of image i and t the number of images. The mean of scene radiance $\hat{\bar{L}}$ can be approximated:

$$\hat{\bar{L}} = \frac{\sum_{i=1}^{t} \sum_{j=1}^{wh} M^{-1} f^{-1}(y_j^i)}{twh},$$

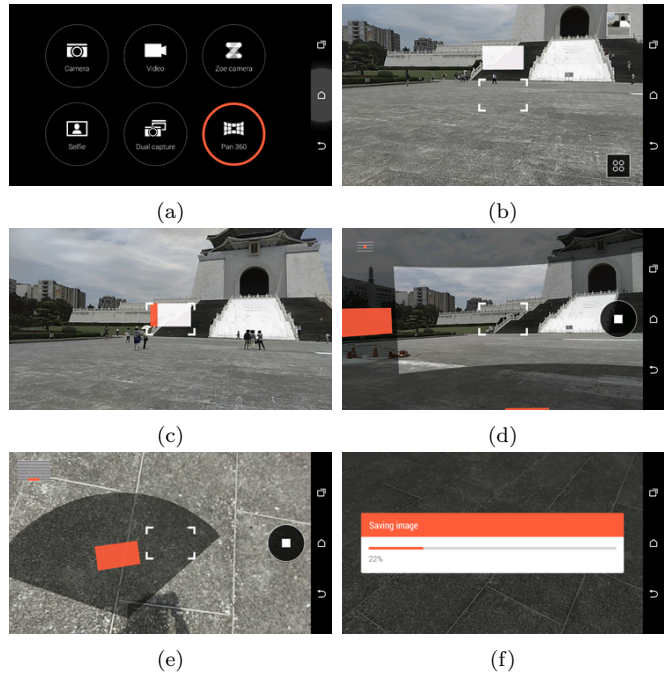
where w is the image width and h the image height.

Initial guess for global exposure k_0^g is calculated by the equation: $k_0^g = \frac{f^{-1}(z(\bar{y}))}{\hat{\bar{L}}}$ and slightly change the value to make sure the estimated overall image tone after global exposure correction falls on a range $[z(\bar{y}) - \epsilon, z(\bar{y}) + \epsilon]$ where ϵ is a constant. With all image color updated, we simply apply linear blending to render the result.

2.4 Immersive Capturing and Viewing

2.4.1 Shooting Target Generation

INS-assisted UI establishes an augmented reality environment for guiding user to easily capture frames and produce high quality panorama. Several virtual rectangles are dynamically generated and equally spread out on the surface of a virtual sphere for user to aim. These rectangle targets guide users to translate and rotate the device to capture targets one after another (Figs. 3(d) and (e)). Once the camera center and indicator completely overlap, the camera captures a photo frame and records the orientation from sensor. Meanwhile, the captured position becomes transparent to notify the user to aim the next target. In addition, a mini-map at the upper-left corner of the screen shows the capturing progress, indicating which rectangles have been filled. The users can stop or capture more rectangles. Once the shooting has completed, Pan360 stitches all captured frames (Fig. 3(f)). We adopt Google PhotoSphere XMP metadata format for cross-platform compatibility.



(a) (b) (c) (d) (e) (f)

Figure 3: Steps of capturing Panorama. (a) User switches to Pan360 mode in the HTC camera app. (b) User aims at a white rectangle. (c) The white rectangle is filled with orange for showing the progress of calibration. (d) When calibration is done, other rectangular indicators appear for next capturing. (e) The last retangle to aim at. (f) Pan360 stitches all captured images.

2.4.2 Random Shooting Sequence

The capturing sequence of the retangles can be in any order. To capture a $360° \times 180°$ panorama, a total number of 44 rectangles are generated. The first frame is forced to be at the center of the scene. After that, capturing the remaining 43 frames can be in any order. Again, a mini-map at the upper-left corner of the screen shows the capturing progress.

2.4.3 Sensor-Guided Panorama Viewing

There are two modes in Pan360: capturing and viewing. In the viewing mode, the panoramic image is projected onto a spherical environment, and the user can navigate the viewpoint by motion sensor, touch gesture, or default motion.

Users can browse panormas in photo gallery (Fig. 4a). If a panoramic photo is selected and the Pan360 icon is tapped, user enters the panorama viewing mode (Fig. 4b) where Pan360 renders a sphere mesh with the panorama mapped onto the sphere (according to the XMP metadata). A mini-map on the upper-left corner indicates the scene coverage. Pan360 supports three browsing ways. First, the view direction is controlled by INS. The downward direction of the sphere is aligned with gravity. Thus, the sphere always locks on the orientation in the real world no matter how the device rotates and translates. Fig. 5 illustrates this scenario. Second, user change view direction by single-finger scrolling, rotate view orientation by two-finger rotation, and zoom by pinch. The default field of view is 60°. The angle of the circular sector on the mip-map changes with zooming (Figs. 4(e) and (f)). Third, Pan360 generates an rotating

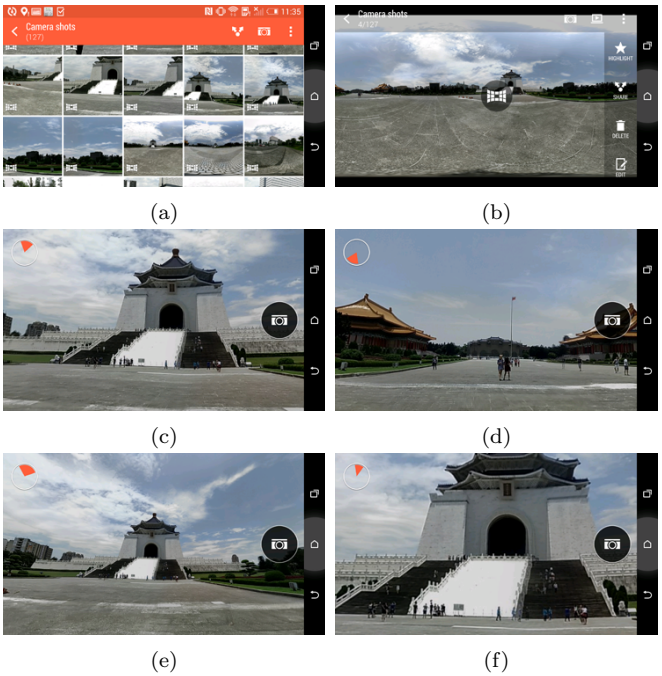


Figure 4: Steps of viewing Panorama. (a) User selects a panorama. (b) User can tap the icon to explore it in 3D immersive view. (c) In virtual view, user can identify the current orientation by the indicator on the upper-left corner. (d) User turns to another direction. (e) User pinches the screen to zoom out. (f) User unpinches the screen to zoom in.

Figure 5: Illustration of the view direction controlled by INS.

animation of the viewing direction. User can take a photo in this mode as if the photo was captured in the scene.

3. RESULTS & CLOSING REMARKS

Fig.6 shows a difficult case for image stitcher, because the texture is insufficient, the structure is repetitive, and even the homography assumption is not valid along the building boundary. We compare stitching result (Fig.6a) with Microsoft ICE [12] (Fig.6b) using full view $2\pi \times \pi$ data. With assistance of inertial sensors, we can deal with the textureless sky region and highly similarity of ground texture. Besides, overall color is well maintained by our method (see the green marked region between Fig.6a and Fig.6b). Fig.6c shows the result of ignoring orientation within MPLSH, and the tiny ghost effect appears in the marked region. The result of disabling multiple matching candidates is in Fig.6d.

Fig.7 shows our vignette correction in color adjustment. With help of the inertial sensors, our method accurately sample the pixels for solving exposure and vignette parameters. The vignette effect in Fig.7c (dark blurry area) is greatly removed in Fig.7d.

Fig.8 shows the comparison of our approach with several commercially mobile applications. We took three panoramas in the same scene for comparison, since those commercial applications only accept image source from camera. The scene is high luminance variation and contains repetitive texture in the ceiling and floor. In the green marked region, our linear blending approach provides appealing result. Our approach also maintain better alignment (see blur marked region). Each application requires a different total number of images (depending on different target-generation UI) to cover the full view of the scene. Our approach requires only 46 images and achieves least capturing time.

Due to the space limitation, more comparison results including different scene structures, light variation and color adjustment for the vignette effect are documented in the extended version [1]. The image source of our experiment data and our customized format for storing image orientations are also provided in the extended version.

Producing good panoramic photos with tranditional cameras is a challenging task, since it requires massive photography and post-production techniques. The high-performance Pan360 not only provides augmented reality UI for guiding users to create high-quality results, but also gives users more impressive viewing experiences via immersive virtual reality. Pan360 was launched in 2014 as an embedded feature of the HTC camera application, and also as a standalone application, PanOMG, at Google Play.

4. REFERENCES

[1] Pan360: Ins assisted 360-degree panorama (extended version). `http://infolab.stanford.edu/~echang/pan360.html`.

[2] Y. Altunbasak, R. M. Mersereau, and A. J. Patti. A fast parametric motion estimation algorithm with illumination and lens distortion correction. *Image Processing, IEEE Transactions on*, 12(4):395–408, 2003.

[3] M. Brown and D. G. Lowe. Automatic panoramic image stitching using invariant features. *International Journal of Computer Vision*, 74(1):59–73, 2007.

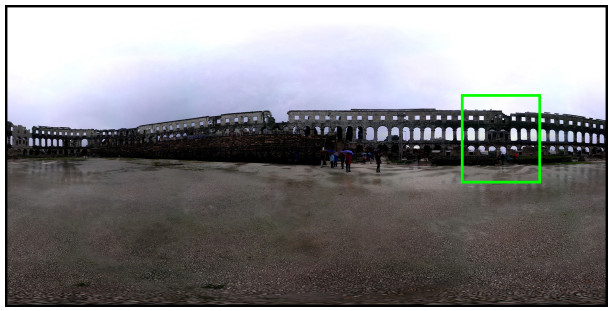
[4] P. J. Burt and E. H. Adelson. A multiresolution spline with application to image mosaics. *ACM Transactions on Graphics (TOG)*, 2(4):217–236, 1983.

[5] P. d'Angelo. Radiometric alignment and vignette calibration. In *Computer Vision Systems, 2007. ICVS 2007. The 5th International Conference on*, 2007.

[6] A. Eden, M. Uyttendaele, and R. Szeliski. Seamless image stitching of scenes with large motions and exposure differences. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2, pages 2498–2505. IEEE, 2006.

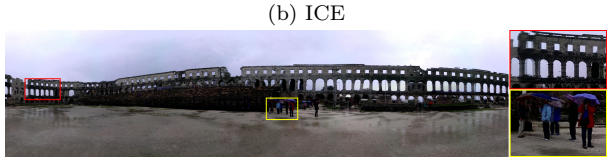
[7] D. B. Goldman and J.-H. Chen. Vignette and exposure calibration and compensation. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 1, pages 899–906. IEEE, 2005.

[8] S. Kim and M. Pollefeys. Robust radiometric calibration and vignetting correction,. In *Pattern Analysis and Machine Intelligence*, volume 30, pages 562–576. IEEE, 2008.

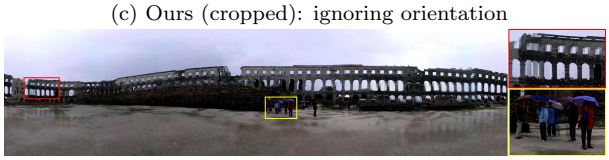


(a) Our result

(b) ICE



(c) Ours (cropped): ignoring orientation

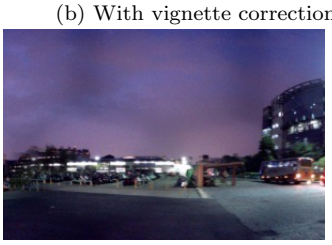


(d) Ours (cropped): ignoring orientation and disable multiple matches

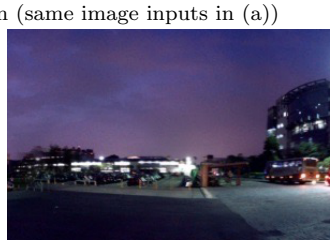
Figure 6: A Difficult Case

(a) Without vignette correction

(b) With vignette correction (same image inputs in (a))



(c) Cropped region of (a)



(d) Cropped region of (b)

Figure 7: Result of Vignette Correcion

(a) Pan360 (Our result. 46 images. Capture time: 90 sec.)

(b) Photo Sphere (Google. 51 images. Capture time: 125 sec.)

(c) Photosynth (Microsoft. 48 images. Capture time: 150 sec.)

Figure 8: Comparison with other applications on mobile device

[9] A. Litvinov and Y. Y. Schechner. Addressing radiometric nonidealities: A unified framework. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 2, pages 52–59. IEEE, 2005.

[10] Q. Lv, W. Josephson, Z. Wang, M. Charikar, and K. Li. Multi-probe lsh: efficient indexing for high-dimensional similarity search. In *Proceedings of the 33rd international conference on Very large data bases*, pages 950–961. VLDB Endowment, 2007.

[11] R. Mantiuk, A. Tomaszewska, and W. Heidrich. Color correction for tone mapping. In *Computer Graphics Forum*, volume 28, pages 193–202. Wiley Online Library, 2009.

[12] Microsoft. Ice. `http://research.microsoft.com/en-us/um/redmond/projects/ice/`, 2015.

[13] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. Orb: an efficient alternative to sift or surf. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2564–2571. IEEE, 2011.

[14] R. Szeliski. *Computer vision: algorithms and applications*. Springer, 2010.

[15] R. Szeliski and H.-Y. Shum. Creating full view panoramic image mosaics and environment maps. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 251–258. ACM Press/Addison-Wesley Publishing Co., 1997.

[16] Y. Xiong and K. Pulli. Color and luminance compensation for mobile panorama construction. In *Proceedings of the international conference on Multimedia*, pages 261–270. ACM, 2010.

[17] Q. Yang, C. Wang, Y. Gao, H. Qu, and E. Y. Chang. Inertial sensors aided image alignment and stitching for panorama on mobile phones. In *Proceedings of the 1st international workshop on Mobile location-based service*, pages 21–30. ACM, 2011.