Aligning Windows of Live Video from an Imprecise Pan-Tilt-Zoom Robotic Camera into a Remote Panoramic Display

Ni Qin[1], Dezhen Song[1], and Ken Goldberg[2]

1: CS Department, Texas A&M University, College Station, TX 77843

2: IEOR and EECS Department, University of California, Berkeley, CA 94720

***Abstract*— A pan-tilt-zoom robotic camera can provide detailed live video of selected areas of interest within a large potential viewing field. To provide spatial context for human observers, it is desirable to insert the resulting live video into a large spherical panoramic display representing the entire viewing field. Accurate alignment of the video stream within the panoramic display is difficult due to small errors in the robot pan-tilt values and image distortion due to nonlinear projection. Existing image alignment algorithms cannot keep up with rapid changes in camera position. In this paper, we present a constant-time image alignment algorithm based on spherical projection and projection-invariant selective sampling that accurately registers paired images at 25 frames per second on a standard PC. Experiments suggest that the new alignment algorithm is faster than previous algorithms by a factor four or more. In a companion paper [1], we present a new calibration algorithm based on image variance density that optimally estimates camera pan-tilt parameters.**

I. Introduction

Scientific study of animals in situ requires vigilant observation of detailed animal behavior over months or years. Since observatories are usually far away from network infrastructure and a stable power supply, they can only be accessed via long distance wireless communication with limited bandwidth. A low-cost, low bandwidth, and energy-efficient solution is to use a tele-operated robotic video camera. Equipped with a high optical zoom lens and pan-tilt mechanism, the camera can track a moving animal with no intrusiveness. To fit the bandwidth constraint and satisfy the responsiveness requirement, the video camera usually transmits a low resolution video (i.e. $\leq 640 \times 480$ pixels) with live frame rate (i.e. > 30 frames per second). Therefore, it suffers from a limited field of view when operated at high zoom levels. An example is the Panasonic HCM 280. When set at a 22x zoom, this camera only covers 2.8° in its horizontal field of view, which looses context of the observed animal behavior.

As illustrated in Figure 1, one way to address the problem is to seamlessly merge the live low resolution video frames into a high resolution panoramic video in real time. Due to the errors introduced by camera potentiometer readings, merging video frames must be based on image registration. Since images usually share the same optical center and only differ in pan-tilt positions, such image registration problems are referred to as the image alignment problem [2].

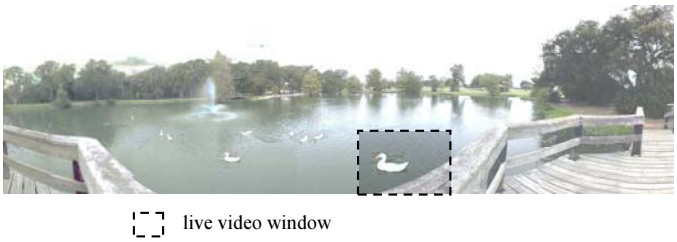


Fig. 1. A live video window is aligned with a panoramic display. It updates the panoramic display as the camera moves.

Existing image alignment algorithms take seconds to align a single image, which does not satisfy the system responsiveness requirement. We found that the alignment computation usually involves two operations: image projection and image matching. The former projects images into alignment space and the later aligns the image by matching intensity values of overlapping pixels or feature points. The computation bottleneck is caused by the coupling between the two operations. A small shift of an image coordinate, which usually is denoted in pan and tilt degrees, changes the image projection matrix and hence changes the shape of the image in the alignment. Therefore, both the projection and the matching have to be repeated for each candidate pan-tilt setting. Since both the projection and the matching are linear to the number of pixels in the aligned image, repeating them extensively slows down the computation.

Our image alignment method builds on the idea of decoupling the projection operations and the matching computations to reduce repetitions in the computation. We achieve this by two techniques. First, we pre-project all images onto a spherical surface. The pre-projection operation significantly reduces the projection distortion in alignment. Therefore, it is possible to sample small square regions from the projected image such that the distortion of each small square region is negligible. We prove that the distortion can be approximated by rotation. We then treat each cell as a rigid object and shift cells to search for the best matching. The resulting constant time algorithm can align frames as fast as 25 frames per

This work was supported in part by the National Science Foundation under IIS-0534848 and IIS-0535218, by Panasonic Corporation, by Intel Corporation, and by UC Berkeley's Center for Information Technology Research in the Interest of Society (CITRIS).

second on a laptop PC and is 4.5x faster than the currently best algorithm.

II. Related Work

Panoramic display is a emerging new way of visualizing remote environments [3]. It involves a variety of research topics including multi-camera systems, omni-directional vision, and panorama construction. Applications exist in many fields such as videoconferencing [4], distance learning [5], tele-operation [6], [7], and natural environment observation [8].

A. Multiple-Camera System and Wide Angle System

When sufficient bandwidth is available, a live panoramic display can be maintained with multiple fixed cameras. Swaminathan and Nayar [9] use four wide angle cameras to monitor a 360° field of view. Similarly, Tan, Hua, and Ahuja [10] combine multiple cameras with a mirror pyramid to create a high resolution panoramic video. Foote et.al. [4], [5] mount five video cameras each with a 3mm lens and near 90° horizontal field of view to provide live omni-panoramic video. When low/variable image resolution is acceptable, a live panoramic display can be maintained with a single wide-angle camera using a fish eye lens or parabolic mirrors [5], [11]–[13].

B. Static Panorama Generation: Image Alignment Techniques

Our live panoramic video system builds on existing research of static panorama construction techniques, which have received a lot of research attention [14]. Panoramas can be classified as either cylindrical panoramas or spherical panoramas according to the number of axes involved in camera motion. A cylindrical panorama only involves pan motion, [15], [16] and its construction is relatively simple and fast. However, cylindrical panoramas cannot provide sufficient vertical field of view for natural environment observation.

Constructing a spherical panorama is much more complex because more parameters needs to be estimated in its nonlinear transformation model. It relies on image alignment techniques, which attempt to find the best set of transform parameters for images to compose the panorama. The transformation can be modeled by a projective projection model [17], [18]. After establishing the parameter model, the image alignment problem searches for a optimized solution in parameter space.

Current image alignment techniques can be classified into three categories: direct method [18]–[23], frequency domain registration [24], [25], and feature-based image registration [26]–[35]. The direct method directly compares intensity values of pixels from the overlapping images and is sensitive to lighting conditions, while feature-based alignment works on a sparse set of feature points and is less sensitive to lighting conditions and needs less computation. Frequency domain registration works well for translation, but has problems with rotation.

Recent research on improving the speed of image alignment focuses on the feature-based method, which extracts features such as Harris corner point [26], [27], [29], Moravec's interest point [30], SUSAN corner point [33], vanishing point [35], and Scale Invariant Feature Transform (SIFT) [36]. Torr and Zisserman [27] outline the feature-based method: First, features are extracted automatically. An initial set of matches are computed based on proximity and similarity of their intensity neighborhood. These estimations inputs are then placed into a robust estimation algorithm such as the Least Median of Squares(LMedS) [28] or Random Sample Consensus(RANSAC) [35] to choose the solution with the largest number of inliers. Numerical minimization techniques such as the Levenberg-Marquardt algorithm are then applied to refine the estimation result from RANSAC.

Our work complements existing research by developing numerical approximation methods. The existing approaches either focus on creating a better and faster feature map [36] or developing a faster algorithm to estimate transformation parameters [27], [28], [35]. Instead, we analyze problem structure and find that the coupling factor between perspective projection and image matching increases the dimensionality of the parameter estimation problem and requires heavy computation. By employing appropriate projections and approximations, we are able to reduce the problem dimensionality and significantly improve on the speed of existing algorithms.

System	Resolution	Bandwidth	Live Motion Images
Our system	Excellent	Low	Yes
Film-based panorama	Excellent	Low	No
Wide-angle systems	Poor	Moderate	Yes
Multi-cameras	Good	Moderate to High	Yes

TABLE I

A comparison of existing methods that can provide panoramic view.

C. Panoramic Video

A high resolution panoramic video can be generated by composing a set of consecutive panoramas. Consecutive panorama can be built from registering a pre-recorded sequence of video frames [37]–[40]. Since conventional image registration based panorama generation suffers from lengthy computation time, existing panoramic video is constructed offline and is referred to as film-based panorama in [5]. Recent progress in feature detection methods such as SIFT [41] further reduces computation time of panorama generation [36]. However, its speed is still much slower than the requirement of *live panoramic video*. Our algorithm satisfies the requirement of live panoramic video on a conventional laptop PC for the first time.

In [5], Foote and Kimber summarize characteristics of different panoramic displays including multiple camera systems, wide angle systems, film-based panorama, and their system using Table I. We expand their table by adding our system for comparison.

III. Problem Definition

A. Assumptions

We assume that all images are taken from a fixed camera which performs pan and tilt movements. Thus all images share the same optical center. Camera potentiometer readings give an estimation of camera pan/tilt position. These readings are

inherently approximate with error (i.e. $\pm 1.5^\circ$) and need to be compensated by image alignment. We assume that the camera intrinsic parameters including image resolution, camera focus length, and CCD sensor size are pre-calibrated and known. We also assume that the camera is a standard video camera with Horizontal Field Of View (HFOV) less than or equal to 45 degrees depending on zoom settings.

B. Inputs and Outputs

The camera is allowed to rotate to different pan-tilt poses to capture images as alignment algorithm input. We use the approximate pan and tilt readings from the camera potentiometer as initial input parameters for the alignment algorithm. Our algorithm outputs accurate camera pan and tilt parameters for real time panorama video construction.

C. Perspective Projection

Image acquisition in a perspective camera is a process that maps a 3D world onto a 2D image plane, which can be described by perspective projection model [14]. We use notations in format of $\{\bullet\}$ to refer to a coordinate system in the paper. Let us define,

- $\{W\}$ as a 3D fixed Cartesian coordinate system with its origin at camera optical center point O. We refer to it as *world coordinate system*. A point in $\{W\}$ is denoted as ${}^{W}Q = {}^{W}[x\ y\ z]^{T}$[1].
- $\{C\}$ as a 3D Cartesian coordinate system with its origin at O, its Z axis overlapping with optical axis, its $X-Y$ plane parallel with CCD sensor plane and its X axis parallel to the horizontal direction of the image. In the paper we refer to it as the *camera coordinate system*. A point in $\{C\}$ is denoted as ${}^{C}Q = {}^{C}[x\ y\ z]^{T}$. Note that $\{C\}$ changes as the camera changes its pan-tilt settings.
- $\{I\}$ as a 2D image plane for image I. The origin of $\{I\}$ is the center of the image. We refer to it as the *image coordinate system*. A point in I is denoted as ${}^{I}q = [u\ v\ 1]^{T}$. In the rest of the paper, we use Q notation to indicate a 3D Cartesian point and q to represent a 2D coordinate.

Therefore, a point in $\{W\}$ is converted to a point in $\{I\}$ by

$$ {}^{I}q = {}^{I}_{C}K\ {}^{C}_{W}R\ {}^{W}Q, \tag{1} $$

where rotation matrix ${}^{C}_{W}R$ maps a point from $\{W\}$ to $\{C\}$ and is determined by camera extrinsic parameters. Intrinsic camera parameters matrix ${}^{I}_{C}K$ projects the points from $\{C\}$ to $\{I\}$. To simplify the notation, we use K instead of ${}^{I}_{C}K$ in the rest of the paper. According to our assumptions, K is a fixed and known matrix.

D. Image Alignment Problem

2D image points in two overlapping images A and B can be mapped with each other using a 3×3 matrix M, [14], [17], [23] as,

$$ {}^{A}q = K\ {}^{A}_{B}R\ K^{-1}\ {}^{B}q = M\ {}^{B}q, \tag{2} $$

where ${}^{A}q$ and ${}^{B}q$ are corresponding points in $\{A\}$ and $\{B\}$, respectively, and rotation matrix ${}^{A}_{B}R$ characterizes the relationship between camera coordinate systems $\{C_A\}$ and $\{C_B\}$ for image A and B, respectively. Since Equation 2 just project pixels in B to $\{A\}$, we refer to the process as the *re-projection* process and M as the *re-projection matrix*.

To align two images we must compute M that minimizes the pixel differences between the two images. Among existing error metrics for pixel differences, Sum of Squared Differences (SSD) is one of the most popular metrics, [14], [22],

$$ SSD = \sum_{i\in A\cap B} \Big(\text{Intensity}_B({}^{B}q_i)) - \text{Intensity}_A({}^{A}q_i)\Big)^2, $$

where set $A\cap B$ is the overlapping pixel set between image A and image B, ${}^{A}q_i$ and ${}^{B}q_i$ are the ith overlapping pixel from image A and image B, respectively, and Intensity_A and Intensity_B are pixel intensity values for image A and B, respectively.

Based on our assumptions, matrix K in Equation 3 is known. M can be determined by camera pan and tilt settings. Therefore, M contains the following independent variables: camera pan angle p_A and p_B for two images, camera tilt angle t_A and t_B for two images,

$$ {}^{A}q = M(p_A, t_A, p_B, t_B)\,{}^{B}q. $$

Assume image B is the newly arrived image, (p_B, t_B) is unknown and (p_A, t_A) were computed when A arrived to the system. M can be determined by two unknown variables (p_B, t_B),

$$ {}^{A}q = M(p_B, t_B)\,{}^{B}q. \tag{3} $$

Therefore, the image alignment problem is to solve the following optimization problem,

$$ \min_{(p_B,t_B)} \sum_{i\in A\cap B} \Big(\text{Intensity}_B(M(p_B,t_B)\,{}^{B}q_i)) - \text{Intensity}_A({}^{A}q_i)\Big)^2. \tag{4} $$

Equation 4 clearly shows the coupling between re-projection (i.e. computing ${}^{A}q_i = M\ {}^{B}q_i$) and intensity comparison. Define $m = |A\cap B|$ as the number of pixels in $A\cap B$ and let k be the number of candidate (p_B, t_B) pairs, a naive search method can easily take $O(km)$ re-projection operations. Since the re-projection computation involves extensive float point computation, km is usually very large and it dominates the overall speed. We are interested in improving the speed by decoupling the re-projection operation and the image matching to reduce factor km.

IV. Algorithm

We use two techniques to decouple the re-projection operation and the image matching, which include a spherical pre-projection and sampling of small regions. First, we pre-project all images onto a spherical surface. The pre-projection significantly reduces the distortion caused by latter re-projection in the alignment phrase. Therefore, it is possible to sample small square regions from the projected image such that the distortion of each small square region is negligible. In the second step of the algorithm, we only need to rotate and shift cells to search for the best matching. We first describe the pre-projection operation that wraps images on a sphere.

[1] We use left superscriptions to indicate the coordinate system of the point.

A. Spherical pre-projection

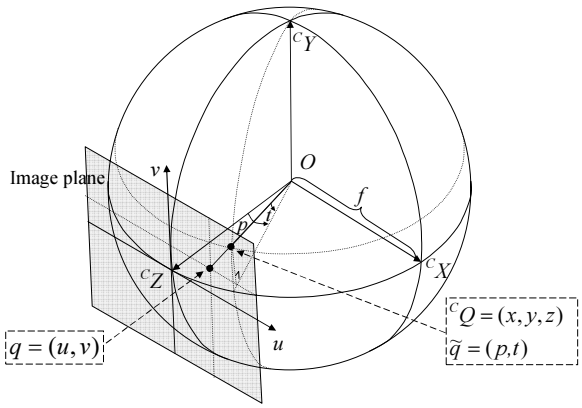


Fig. 2. An illustration of spherical pre-projection and coordinate systems: q in image coordinate system, $\tilde{q}$ on the local spherical coordinate system, and ${}^{C}Q$ is the same point as $\tilde{q}$ but in the camera coordinate system.

Recall that I is the image captured by the camera. It is first projected onto the surface of a sphere that is centered at the camera optical center and has a radius the same as focal length f. As illustrated in Figure 2, the projection generates a wrapped image $\tilde{I}$ based on a *local spherical coordinate system* $\{\tilde{I}\}$. Let us define:

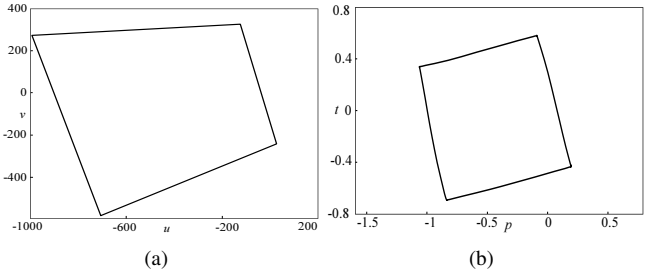
- $q = (u, v)^T$ as a point in I.
- $\tilde{q} = (p, t)^T$ as the corresponding point in $\tilde{I}$.

The spherical pre-projection that projects q to $\tilde{q}$ is,

$$p = \arctan(\frac{u}{f}), \tag{5a}$$

$$t = -\arctan(\frac{v}{\sqrt{u^2 + f^2}}). \tag{5b}$$

Each point in $\tilde{I}$ is defined using local pan and tilt spherical coordinates with units in radians. Spherical coordinate system $\{\tilde{I}\}$ usually consists of three elements including radius, pan, and tilt. Since image $\tilde{I}$ is on the spherical surface with the same radius as f for all pixels, we omit it and yield a 2D representation. Also, $\tilde{q} = (0, 0)^T$ overlaps with $q = (0, 0)^T$. Note that $\{\tilde{I}\}$ is centered at each image and is different from the global spherical coordinate defined by real camera pan and tilt settings.



(a) (b)

Fig. 3. Comparison of image distortion caused by the re-projection operation (a) in the original image space and (b) on the spherical surface. In this example, we perform perspective projection computation to map a rectangular image to another new location that shares 30° tilt value and has 30° pan difference.

We know that images must be in the same coordinate system before computing the SSD. Re-projection matrix M in Equation 4 projects image B into $\{A\}$. A similar re-projection operation should be performed after the spherical pre-projection and before computing SSD. We are interested in discovering if the amount of distortion introduced by the re-projection process will be different before and after spherical pre-projection. Figure 3 suggests that the distortion in the spherical surface is significantly less than that in the original image space. Since the absolute distortion is an increasing function of image size, we conjecture that if we sample a very small square region on the spherical surface, which is named as *cell*, the distortion for each cell should be negligible after the spherical pre-projection. If so, it is possible to reduce the re-projection cost in image alignment.

B. Distortion Invariant in Cell Re-projection

To prove the conjecture, we explore the re-projection process using both the local spherical coordinate system and the camera coordinate system. Define $Q = {}^{C}Q = [x, y, z]^T$ as $\tilde{q}$ in $\{C\}$ as illustrated in Figure 2. For simplification, $\cos(\theta)$ and $\sin(\theta)$ are denoted as $c(\theta)$ and $s(\theta)$, respectively. The relationship between $\{\tilde{I}\}$ and $\{C\}$ can be described by function P and its inverse P^{-1},

$$\tilde{q} = \begin{bmatrix} p \\ t \end{bmatrix} = \begin{bmatrix} \arctan(x/z) \\ -\arctan(y/\sqrt{x^2 + z^2}) \end{bmatrix} = P(Q), \tag{6}$$

$$Q = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} f \cdot c(t)s(p) \\ -f \cdot s(t) \\ f \cdot c(t)c(p) \end{bmatrix} = P^{-1}(\tilde{q}). \tag{7}$$

Let $\tilde{A}$ and $\tilde{B}$ be the resulting image from the spherical pre-projection for image A and image B, respectively. Without loss of generality, we select image $\tilde{A}$ as the reference image. We shift image $\tilde{B}$ around $\tilde{A}$. To align the two images, we need to re-project $\tilde{B}$ into $\tilde{A}$'s space,

$$\begin{aligned} {}^{A}\tilde{q} &= P({}^{A}_{B}R\ {}^{B}Q) = P({}^{A}_{B}R\ P^{-1}({}^{B}\tilde{q})) \\ &= F({}^{A}_{B}R, {}^{B}\tilde{q}). \end{aligned} \tag{8}$$

where F is the re-projection function.

Equation 8 suggests that F is a nonlinear function and may contain significant nonlinear distortions. However, Figure 3(b) suggests that the distortion of a small cell under F may be negligible. Let us define a squared-shaped cell in image $\tilde{A}$ as,

$${}^{A}\mathbb{C} = \{({}^{A}p, {}^{A}t) | {}^{A}p \in [{}^{A}p_o \pm p_c], {}^{A}t \in [{}^{A}t_o \pm t_c]\},$$

where $({}^{A}p_o, {}^{A}t_o)$ is the cell center coordinate, and (p_c, t_c) is the maximum cell span in pan and tilt directions. Similarly, we define ${}^{B}\mathbb{C}$ as a cell in image $\tilde{B}$. For a small cell, we have the following lemma.

Lemma 1: If the spherical cell is small ($p_c \leq 5°$ and $t_c \leq 5°$), define point $({}^{B}\tilde{q}_o + \mathbf{\Delta}{}^{B}q)) \in {}^{B}\mathbb{C}$ and its corresponding point $({}^{A}\tilde{q}_o + \mathbf{\Delta}{}^{A}q)) \in \tilde{A}$,

$$F({}^{A}_{B}R, ({}^{B}\tilde{q}_o + \mathbf{\Delta}{}^{B}\tilde{q})) = {}^{A}\tilde{q}_o + \Delta{}^{A}\tilde{q}, \tag{9}$$

we have

$$\mathbf{\Delta}{}^{A}\tilde{q} \approx R_c \mathbf{\Delta}{}^{B}\tilde{q}, \tag{10}$$

where R_c is a 2×2 rotation matrix. This means the cell distortion under re-projection defined in Equation 8 is negligible.

Proof: Let us expand our notations in detail.

- ${}^A\tilde{q}_o = [{}^Ap_o, {}^At_o]^T$ and ${}^B\tilde{q}_o = [{}^Bp_o, {}^Bt_o]^T$,
- $\mathbf{\Delta}^A\tilde{q} = [\Delta^Ap, \Delta^At]^T$ and $\mathbf{\Delta}^B\tilde{q} = [\Delta^Bp, \Delta^Bt]^T$.

For vector calculus, we know that

$$\nabla Q = \begin{bmatrix} fc(t)c(p) & -fs(t)s(p) & c(t)s(p) \\ 0 & -fc(t) & -s(t) \\ -fc(t)s(p) & -fs(t)c(p) & c(t)c(p) \end{bmatrix} \begin{bmatrix} dp \\ dt \\ df \end{bmatrix} \quad (11)$$

Therefore, we have

$$\begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \end{bmatrix} = f \begin{bmatrix} c(t)c(p) & -s(t)s(p) & m_{13} \\ 0 & -c(t) & -m_{23} \\ -c(t)s(p) & -s(t)c(p) & m_{33} \end{bmatrix} \begin{bmatrix} \Delta p \\ \Delta t \\ \Delta f \end{bmatrix}, \quad (12)$$

where $m_{13} = c(t)c(p)/f$, $m_{23} = s(t)/f$, $m_{33} = c(t)c(p)/f$ corresponds to the last column of the Jacobian matrix in Equation 11, $[\Delta x, \Delta y, \Delta z]^T$ is the small displacement in $\{C\}$, and $[\Delta p, \Delta t, \Delta f]^T$ is the corresponding change in $\{\tilde{I}\}$. Since we have $\{\tilde{I}\}$ as part of a sphere, radius f remains constant. Therefore $\Delta f = 0$. To move the negative sign out of the second row of the matrix in Equation 12, we introduce coefficient matrix H,

$$H = \begin{bmatrix} f & 0 & 0 \\ 0 & -f & 0 \\ 0 & 0 & f \end{bmatrix}.$$

Then Equation 12 can be rewritten as,

$$\begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \end{bmatrix} = H \begin{bmatrix} c(t)c(p) & -s(t)s(p) & m_{13} \\ 0 & c(t) & m_{23} \\ -c(t)s(p) & -s(t)c(p) & m_{33} \end{bmatrix} \begin{bmatrix} \Delta p \\ \Delta t \\ 0 \end{bmatrix}, \quad (13)$$

Recall that t are the tilt positions with respect to the image center inside an image. A standard camera has a maximum horizontal field of view of 45° and a maximum vertical field of view of 34°. The maximum value of t is $34/2 = 17^\circ$. Since $\cos(17^\circ) = 0.956$, therefore, $0.956 \leq c(t) \leq 1$. The maximum distortion is less than 5%, which is less than half a pixel for a cell size of 20×20 pixels. Since the distortion is very small, instead we drop $c(t)$ in the first column,

$$\begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \end{bmatrix} \approx H \begin{bmatrix} c(p) & s(p)s(-t) & m_{13} \\ 0 & c(-t) & m_{23} \\ -s(p) & c(p)s(-t) & m_{33} \end{bmatrix} \begin{bmatrix} \Delta p \\ \Delta t \\ 0 \end{bmatrix}, \quad (14)$$

Since $\Delta f = 0$, we know that $[m_{13}, m_{23}, m_{33}]^T$ can take arbitrary values without affecting the equality in Equation 14. Let us choose $m_{13} = s(p)c(-t)$, $m_{23} = -s(-t)$, and $m_{33} = c(p)c(-t)$. Then we have,

$$\begin{bmatrix} c(p) & s(p)s(-t) & s(p)c(-t) \\ 0 & c(-t) & -s(-t) \\ -s(p) & c(p)s(-t) & c(p)c(-t) \end{bmatrix} = R_Y(p)R_X(-t), \quad (15)$$

where R_Y and R_X are rotation matrices along Y axis and X axis, respectively. Define $\Delta Q = [\Delta x, \Delta y, \Delta Z]^T$ and $\Delta\tilde{q} = [\Delta p, \Delta t, 0]^T$, Now Equation 14 is,

$$\Delta Q \approx HR_Y(p)R_X(-t)\Delta\tilde{q} \quad (16)$$

Hence, we have

$$\Delta^A Q \approx HR_Y({}^Ap_o)R_X(-{}^At_o) \begin{bmatrix} \Delta^A\tilde{q} \\ 0 \end{bmatrix}, \quad (17)$$

and,

$$\Delta^B Q \approx HR_Y({}^Bp_o)R_X(-{}^Bt_o) \begin{bmatrix} \Delta^B\tilde{q} \\ 0 \end{bmatrix}. \quad (18)$$

Since $\Delta^A Q = {}^A_B R \Delta^B Q$, we get,

$$\begin{bmatrix} \Delta^A\tilde{q} \\ 0 \end{bmatrix} \approx R_X({}^At_o)R_Y(-{}^Ap_o)H^{-1} \cdot {}^A_B RHR_Y({}^Bp_o)R_X(-{}^Bt_o) \begin{bmatrix} \Delta^B\tilde{q} \\ 0 \end{bmatrix}. \quad (19)$$

Since H and H^{-1} are diagonal matrices, we have $H^{-1}{}^A_B RH = {}^A_B R$. Equation 19 becomes,

$$\begin{bmatrix} \Delta^A\tilde{q} \\ 0 \end{bmatrix} \approx R_\Delta \begin{bmatrix} \Delta^B\tilde{q} \\ 0 \end{bmatrix}, \quad (20)$$

where

$$R_\Delta = R_X({}^At_o)R_Y(-{}^Ap_o){}^A_B RR_Y({}^Bp_o)R_X(-{}^Bt_o), \quad (21)$$

is a rotation matrix because the multiplication of rotation matrices yields a rotation matrix. On the other hand, the last row has to satisfy $0 = 0$ no matter what value $\Delta^B\tilde{q}$ takes. This means R_Δ has to be in the following format,

$$R_\Delta = \begin{bmatrix} R_c & 0_{2\times 1} \\ 0_{1\times 2} & 1 \end{bmatrix},$$

where R_c is a 2×2 rotation matrix. ■

Lemma 1 suggests that each cell can be treated as a rigid object in the projection, which could lead to a significant computation savings when compared to methods that directly use Equation 8. The next question is how to compute the rotation matrix R_c, which can be characterized by a single rotation angle θ. We have the following lemma,

Lemma 2: Recall that (p_A, t_A) and (p_B, t_B) are the pan and tilt settings for image A and B, respectively. Rotation angle θ of rotation matrix R_c can be approximated by,

$$\theta \approx \arccos\Big(c({}^Ap_o)c({}^Bp_o)c(p_B - p_A) + s({}^Ap_o)s({}^Bp_o) * \alpha + s(p_B - p_A)s({}^Ap_o)c({}^Bp_o)c(t_A) - s(p_B - p_A)c({}^Ap_o)s({}^Bp_o)c(t_B)\Big). \quad (22)$$

where α is a function of (p_A, t_A) and (p_B, t_B) only and can be pre-computed.

$$\alpha = c(t_A)c(t_B)c(p_B - p_A) + s(t_A)s(t_B). \quad (23)$$

(α is the dot product of Z axis of $\{C_A\}$ and $\{C_B\}$ in world coordinate system.)

Proof: Let us use the following vectors,

- $\begin{bmatrix} \Delta^A\tilde{q} \\ 0 \end{bmatrix} = [1/f, 0, 0]^T$,
- $\begin{bmatrix} \Delta^B\tilde{q} \\ 0 \end{bmatrix} = [1/f, 0, 0]^T$,
- ${}^AX_{0A} = HR_Y({}^Ap_o)R_X(-{}^At_o) \begin{bmatrix} \Delta^A\tilde{q} \\ 0 \end{bmatrix}$, and

- ${}^{B}X_{0B} = HR_Y({}^{B}p_o)R_X(-{}^{B}t_o)\begin{bmatrix} \Delta^{B}\tilde{q} \\ 0 \end{bmatrix}$.

It is clear that ${}^{A}X_{0A}$ and ${}^{B}X_{0B}$ are unit vectors. By defining ${}^{W}X_{0A}$ and ${}^{W}X_{0B}$ as their corresponding coordinate in $\{W\}$, we know that

$$c(\theta) =< {}^{W}X_{0A}, {}^{W}X_{0B} >, \tag{24}$$

from the definition of vector inner product. From coordinate transform relationship, we know,

$$\begin{aligned} {}^{W}X_{0A} &= {}^{W}_{C_A}R\ {}^{A}X_{0A} \\ &= R_Y(p_A)R_X(t_A)HR_Y({}^{A}p_o)R_X(-{}^{A}t_o)[1/f,0,0]^T \\ &= \begin{bmatrix} c(p_A)c({}^{A}p_o) - s(p_A)c(t_A)s({}^{A}p_o) \\ s(t_A)s({}^{A}p_o) \\ -s(p_A)c({}^{A}p_o) - c(p_A)c(t_A)s({}^{A}p_o) \end{bmatrix}. \end{aligned}$$

Inserting them into Equation 24, we get Equation 22. ■

Remark It is worth mentioning that if two images share similar pan positions (i.e. $|p_A - p_B| \leq 5^\circ$), then Equation 22 becomes

$$\theta \approx \arccos\Big(c({}^{A}p_o)c({}^{B}p_o) + s({}^{A}p_o)s({}^{B}p_o)c(t_B - t_A)\Big). \tag{25}$$

Recall that a standard camera has a maximum vertical field view of 34°. To guarantee the overlap between the two frames, the maximum value of $t_B - t_A$ has to be less than 17°. Therefore, $0.956 \leq c(t_B - t_A) \leq 1$ and $c(t_B - t_A)$ can be approximated by 1. Hence, we have,

$$\theta \approx {}^{B}p_o - {}^{A}p_o,$$

for this special case, which can further speed up the computation.

Based on the analysis, we develop a *Cell-Based Image Alignment Algorithm*.

C. Cell-Based Image Alignment Algorithm

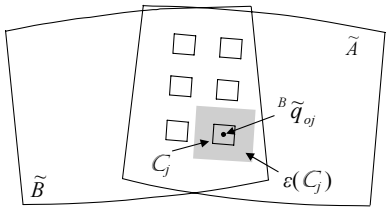


Fig. 4. An illustration of Cell-Based Image Alignment Algorithm. Image $\tilde{A}$ and image $\tilde{B}$'s barrel-like shape is due to spherical pre-projection.

As illustrated in Figure 4, our algorithm is based on a set of small square-shaped cells scattered in the overlapping region. Define k_c as the number of cells, which is between 25 and 36 in most cases. Define $\mathbb{C}_j \subset \tilde{A}$, $1 \leq j \leq k_c$ as the jth cell. From potentiometer reading and its error range, we know that the matching region of $F^{-1}(\mathbb{C}_j) \subset \tilde{B}$ will be found within region $\epsilon(\mathbb{C}_j) \subset \tilde{A}$, which is the gray region in Figure 4,

$$\begin{aligned} \epsilon(\mathbb{C}_j) = \{(p,t) \in \tilde{A} | p \in [{}^{A}p_{oj} \pm (p_c + .5p_{\max})] \\ t \in [{}^{A}t_{oj} \pm (t_c + .5t_{\max})]\}, \end{aligned} \tag{26}$$

where $({}^{A}p_{oj}, {}^{A}t_{oj}) = {}^{A}\tilde{q}_{oj}$ is the center point of $\mathbb{C}_j$, (p_c, t_c) defines cell size, and $(p_{\max}, t_{\max})$ is the potentiometer error range. For example, for the images captured from a Canon VCC3 camera that has a 45° horizontal field of view, an image size of 640×480-pixels, and $\pm 1.5^\circ$ potentiometer error, $\epsilon(\mathbb{C}_j)$ is ± 20 pixels shifting range in $\tilde{A}$. We also know that the inverse projection $F^{-1}(\mathbb{C}_j)$ is the inverse rotation by $-\theta$ around cell center ${}^{B}q_{oj}$,

$$F^{-1}(\mathbb{C}_j) = R_c(-\theta)\mathbb{C}_j.$$

Therefore, we transfer the optimization problem in Equation 4 to

$$\min_{(p_B, t_B)} \sum_{j=1}^{k_c} \Big(\text{Intensity}_B(R_c(-\theta)\mathbb{C}_j) - \text{Intensity}_A(\mathbb{C}_j)\Big)^2, \tag{27}$$

subject to,

$$\mathbb{C}_j \subset \epsilon(\mathbb{C}_j). \tag{28}$$

Since $\mathbb{C}_j$ is considered as a solid square with only rotation and shifting, computing the solution becomes less costly. Each candidate solution will determine orientation and location of k_c cells $R_c(-\theta)\mathbb{C}_j \subset \tilde{B}$, $j = 1, ..., k_c$.

Since the relative position between cells are rigid and known, the search for a solution is to simultaneously shift all k_c rotated cells in $\tilde{A}$ and find the optimal solution with the pre-computed $R_c(-\theta)\mathbb{C}_j$'s. Because k_c is a relatively small number (i.e. $25 \sim 36$) and each cell is very small (i.e. 10×10 pixels), the computation is very fast. Define $(\delta p_A, \delta t_A)$ as $\mathbb{C}_j$ shifting variable such that $\delta p_A \in [\pm 0.5 p_{\max}]$ and $\delta t_A \in [\pm 0.5 t_{\max}]$ to satisfy Equation 28. Because of the image resolution limit, there are only a constant number of $(\delta p_B, \delta t_B)$ pairs.

Another benefit is that feature detection and spherical pre-projection do not need to be computed for the entire image. Only pixels in the selected cells and their neighboring search regions need to be computed. Define n as the number of pixels in image A and image B. Restricting the feature detection range from the entire image to a fix number that is determined by constant k_c further cuts the running time from $O(n)$ to $O(1)$. We summary the analysis above as the Cell-Based Image Alignment Algorithm below.

Cell Based Image Alignment Algorithm

Select k_c cells from the overlapping region in $\tilde{A}$.	$O(1)$
Sphere projection	$O(1)$
Feature detection in the cell and searching regions.	$O(1)$
For each $(\delta p_B, \delta t_B)$	$O(1)$
For each cell	$O(1)$
Compute $\mathbb{C}_j$	
(*) Compute $R_c(-\theta)\mathbb{C}_j \subset \tilde{B}$, $j = 1, ..., k_c$.	
Compute SSD between $R_c^{-1}\mathbb{C}_j \subset \tilde{B}$ and $\mathbb{C}_j \subset \tilde{A}$	
End For	
Report sum of SSD across all cells	
End For	
Output solution with the minimum SSD.	

Therefore, we have,

Theorem 1: For an n-pixel image pair, the cell-based image alignment algorithm runs in constant time.

Remark Step (*) in Cell Based Image Alignment Algorithm is in the For loop. It can be pre-computed using a table based on $(\delta p_B, \delta t_B)$, which will further improve the algorithm speed.

V. Experiments and Results

We have implemented the algorithm and tested in-situ. The computer we used for testing is a 1.6Ghz Centrino laptop PC with 512RAM and 40GB hard disk.

A. Speed Test

We first compare the speed of our algorithm with the fastest currently available method in [36]. We downloaded their autostitch program from their website and did a comparison using 21 images captured on the UC Berkeley campus as testing data. Images are captured by a Canon VCC3 camera with a 45° horizontal field of view. Each image has a resolution of 320×240. Our algorithm aligns all 21 images in 881 miliseconds while the autostitch program in [36] needs 4 seconds. We are able to reach a frame rate of around 25 frames per second on a conventional laptop PC. Our algorithm is 4.5x faster than the autostitch program at a resolution of 320×240.

B. Field Tests

We are currently working with natural scientists to identify testing sites. We will report more results in the future. Meanwhile, we are testing our systems in parks near networking infrastructure. Figure 5 illustrates snapshot of live panoramic videos generated during bird watching. Experiments were conducted from in both College Station, Texas and Richardson Bay Audubon Sanctuary in San Francisco Bay. We have collected 2186 frames and the original panorama has a resolution of 4000×1000. The camera used is a Panasonic HCM 280 networked pan-tilt-zoom camera.

VI. Conclusion and Future Work

We propose a constant time algorithm to compose live panoramic video to visualize remote natural environments using a tele-operated robotic camera. Our algorithm seamlessly aligns live video frames into a panoramic display in real time. Existing alignment algorithms involves extensive computation and cannot satisfy real time requirements. Our analysis shows that the computational bottleneck in existing algorithms is the repetitive computation of image projection and image matching. To address the problem, we pre-project images onto a spherical surface. The resulting constant time algorithm can align frames as fast as 25 frames per second on a laptop PC. Experiments show that the new algorithm is 4.5x faster than the best algorithm available.

In the future, we will further improve the algorithm by effectively choosing the location of each cell and/or weight cells differently to improve performance without sacrificing speed. To process the large amount of spatial video data, we will develop new protocols and data structures for fast data transmission, query, and storage. We will also consider adding sensor inputs and a image processing module to automatically drive the camera to capture interesting events.

Acknowledgments

Special thanks are give to E. Brewer for providing camera installation site. Thanks are given to Q. Hu, Z. Goodwin, and T. Schmidt for implementing part of the system. Our thanks to M. Pantaleano, A. Parish, A. Coots, K. Jaehan, N. Amato, T. Lao, D. Volz, T. Ioerger, R. Gutierrez-Osuna, V. Taylor for insightful discussions and feedback.

References

[1] D. Song, N. Qin, and K. Goldberg, "A minimum variance calibration algorithm for pan-tilt robotic cameras in natural environments," in *IEEE International Conference on Robotics and Automation (ICRA), Orlando, Florida*, May 2006.

[2] B. Zitova and J. Flusser, "Image registration methods: a survey," *Image and Vision Computing*, vol. 21, no. 2003, pp. 977–1000, June 2003.

[3] R. Benosman and S. B. Kang, *Panoramic Vision*. Springer, New York, 2001.

[4] J. Foote and D. Kimber, "Flycam: Practical panoramic video and automatic camera control," in *IEEE International Conference on Multimedia and Expo. ICME 2000, New York, NY*, vol. 3, July 2000, pp. 1419–1422.

[5] ——, "Enhancing distance learning with panoramic video," in *Proceedings of the 34th Hawaii International Conference on System Sciences*, 2001.

[6] D. Song and K. Goldberg, "Sharecam part I: Interface, system architecture, and implementation of a collaboratively controlled robotic webcam," in *IEEE/RSJ International Conference on Intelligent Robots (IROS)*, Nov. 2003.

[7] D. Song, A. Pashkevich, and K. Goldberg, "Sharecam part II: Approximate and distributed algorithms for a collaboratively controlled robotic webcam," in *IEEE/RSJ International Conference on Intelligent Robots (IROS)*, Nov. 2003.

[8] D. Song and K. Goldberg, "Networked robotic cameras for collaborative observation of natural environments," in *The 12th International Symposium of Robotics Research, (ISRR), San Francisco, CA*, October 2005.

[9] R. Swaminathan and S. K. Nayar, "Nonmetric calibration of wide-angle lenses and polycameras," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 10, pp. 1172–1178, October 2000.

[10] K.-H. Tan, H. Hua, and A. N., "Multiview panoramic cameras using mirror pyramids," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 7, pp. 1941– 946, July 2004.

[11] S. Baker and S. K. Nayar, "A theory of single-viewpoint catadioptric image formation," *International Journal of Computer Vision*, vol. 35, no. 2, pp. 175 – 196, November 1999.

[12] S. K. Nayar, "Catadioptric omnidirectional camera," in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, San Juan, Puerto Rico*, June 1997, pp. 482–488.

[13] Y. Xiong and K. Turkowski, "Creating image-based vr using a self-calibrating fisheye lens," in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, San Juan, Puerto Rico*, June 1997, pp. 237–243.

[14] R. Szeliski, "Image alignment and stitching, microsoft research, technical report msr-tr-2004-92," 2004.

[15] S. E. Chen, "QuickTime VR — an image-based approach to virtual environment navigation," *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1995, Los angeles, California*, vol. 29, pp. 29–38, Aug 1995.

[16] B. Y. Kim, K. H. Jang, and S. K. Jung, "Adaptive strip compression for panorama video streaming," in *Computer Graphics International (CGI'04), Crete, Greece*, June 2004.

[17] R. Hartley, "Self-calibration of stationary cameras," in *IJCV*, vol. 22, 1997, pp. 5–23.

[18] H. Shum and R. Szeliski, "Panoramic image mosaics," 1997, microsoft Research:MSR-TR-97-23.

[19] S. Coorg and S. Teller, "Spherical mosaics with quaternions and dense correlation," *International Journal of Computer Vision*, vol. 37, no. 3, pp. 259–273, 2000.

[20] S. B. Kang and R. Weiss, "Characterization of errors in compositing panoramic images," in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, San Juan, Puerto Rico*, June 1997, pp. 103–109.

[21] R. Szeliski, "Image mosaicing for tele-reality applications," in *Proceedings of the Second IEEE Workshop on Applications of Computer Vision, Sarasota, USA*, Dec 1994, pp. 44–53.

(a)

(b)

Fig. 5. Snapshots of panoramic videos created for bird watching in (a) Central Park, College Station, TX in Auguest 24, 2005 and (b) Richardson Bay Audubon Sanctuary in San Francisco Bay, CA in Dec. 21, 2005.

[22] ——, "Video mosaics for virtual environments," in *Proceedings of IEEE Computer Graphics and Applications*, vol. 16, no. 2, Mar 1996, pp. 22–30.

[23] R. Szeliski and H. Shum, "Creating full view panoramic image mosaics and environment maps," *Proceedings of the 24th annual conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1997, Los Angeles, California*, vol. 31, pp. 251–258, Aug 1997.

[24] E. D. Castro and C. Morandi, "Registration of translated and rotated images using finite fourier transform," in *Proceedings of IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 9, no. 5, 1987, pp. 700–703.

[25] B. S. Reddy and B. N. Chatterji, "An fft-based technique for translation, rotation, and scale-invariant image registration," in *Proceedings of IEEE Transactions on Image Processing*, vol. 5, no. 8, Aug 1996, pp. 1266–1271.

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

[27] P. H. S. Torr and A. Zisserman, "Feature based methods for structure and motion estimation," in *Proceedings of the International Workshop on Vision Algorithm: Theory and Practice, Corfu*, 1999, pp. 278–294.

[28] Z. Zhang, R. Deriche, O. Faugeras, and Q. T. Luong, "A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry," *Artificial Intelligence*, vol. 78, pp. 87–119, 1995.

[29] I. Zoghlami, O. Faugeras, and R. Deriche, "Using geometric corners to build a 2d mosaic from a set of images," in *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition, San Juan, Puerto Rico*, June 1997, pp. 420–425.

[30] B. Hu, C. Brown, and A. Choi, "Acquiring an environment map through image mosaicking," in *University of Rochester:TR-786*, Nov 2001.

[31] G. Borgefors, "Hierarchical chamfer matching: a parametric edge matching algorithm," in *Proceedings of IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 10, no. 6, Nov 1988, pp. 849–865.

[32] H. Li, B. S. Manjunath, and S. K. Mitra, "A contour-based approach to multisensor image registration," in *Proceedings of IEEE Transactions on Image Processing*, vol. 4, no. 3, Mar 1995, pp. 320–334.

[33] S.-H. Cho, Y.-K. Chung, and J. Y. Lee, "Automatic image mosaic system using image feature detection and taylor series," in *Proceedings of VIIth Digital Image Computing: Techniques and Applications. Sydney*, Dec. 2003, pp. 549–556.

[34] Y. Kanazawa and K. Kanatani, "Image mosaicing by stratified matching," in *Proceedings of Statistical Methods in Video Processing Workshop, Denmark*, Jan. 2002, pp. 31–36.

[35] W. Zhang, J. Kosecka, and F. Li, "Mosaics construction from a sparse set of views," in *Proceedings of First International Symposium on 3D Data Processing Visualization and Transmission*, June 2002, pp. 177–180.

[36] M. Brown and D. Lowe, "Recognising panoramas," in *Proceedings of IEEE International Conference on Computer Vision, Nice, France*, vol. 2, Oct 2003, pp. 1218–1225.

[37] M. Irani, P. Anandan, J. Bergen, R. Kumar, and S. Hsu, "Mosaic representations of video sequences and their applications," in *Signal Processing: Image Communication*, vol. 8, May 1996, pp. 327–351.

[38] E. Trucco, A. Doull, F. Odone, A. Fusiello, and D. Lane, "Dynamic video mosaicing and augmented reality for subsea inspection and monitoring," Mar 2000.

[39] Z. Zhu, G. Xu, E. M. Riseman, and A. R. Hanson, "Fast generation of dynamic and multi-resolution 360-degree panorama from video sequences," in *Proceedings of the IEEE International Conference on Multimedia Computing and Systems, Florence, Italy*, vol. 1, June 1999, pp. 9400–9406.

[40] A. Agarwala, C. Zheng, C. Pal, M. Agrawala, M. Cohen, B. Curless, D. Salesin, and R. Szeliski, "Panoramic video textures," in *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2005), Los Angeles, CA*, July 2005.

[41] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proceedings of 7th IEEE International Conference on Computer Vision, Corfu, Greece*, vol. 2, Sept 1999, pp. 1150–1157.