Real Time View Alignment between Widely Spaced Cameras

Master of Science Project Report

Gaurav Chanda

GVU Lab

College of Computing

Georgia Institute of Technology

gchanda@cc.gatech.edu

Abstract

View alignment between widely spaced cameras for a general scene is a hard problem in computer vision. However, in a meeting room scenario, certain assumptions can be made and the problem can be made more tractable. In this project, a system has been designed which enhances the view of an omnidirectional camera placed at the center of the room with views obtained from other cameras in the room. Important features are detected and matched and a warping function is computed in real time.

1 Introduction

Image registration from multiple sensors has been an important problem for many years. The problem has applications in computer vision and graphics, medical image analysis and processing of data obtained from remote sensors like satellite imagery. In computer vision, image registration helps in solving the correspondence problem which is fundamental to many applications like stereo reconstruction

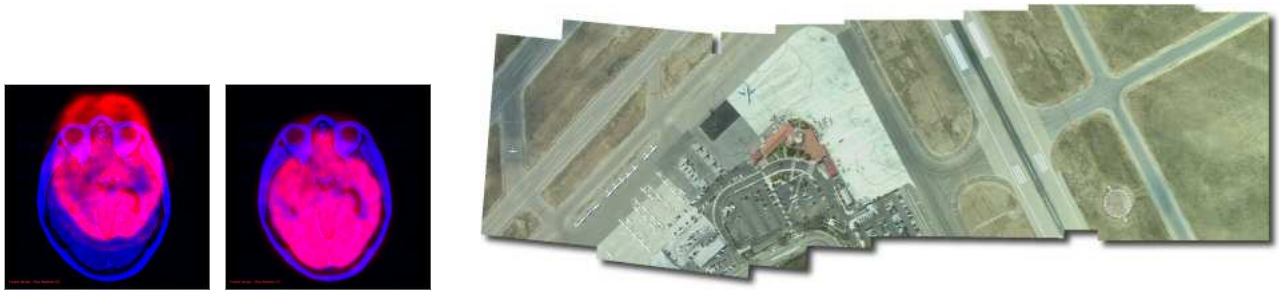

Figure 1: Examples of applications of image registration : *Medical Imagery* registering 2 sources of data (blue and red). *Satellite Imagery* registering sequence of images taken from satellite.

[Merickel, 1988], optical flow computation for motion tracking and segmentation [Guillox, 1986] etc. It is also useful in computer graphics for image based rendering applications [Uyttendaele et al., 2003]. Using a small set of views from different vantage points, the entire scene can be represented and navigated by a user by aligning the views among each other. In medical imagery, registration of CAT scans helps in detecting tumors and cancerous tissues [Pelizzari et al., 1989]. Aligning satellite images taken at different points in time (say, a few decades apart), helps in studying changes in the topology of cities. Registration of images obtained from other satellite sensors is also used in agriculture, oceanography and locating new sources of oil and natural gas.

View alignment is the problem of registering 2 images of the same scene taken from different cameras. The output of alignment is a transformation function, which gives for every pixel in the first image, its corresponding position in the second image. The transformation functions can vary from simple linear functions (rotation, translation and shear) to arbitrarily complicated functions (elastic model based matching) [Ratib et al., 1988] depending on the scene assumptions and camera placements. These functions can either be global/local, again depending on whether the computation needed to take every patch/point can be represented by one (possible complicated) function or whether the computation is only piecewise smooth. Transformation can also be a combination of one global function followed by local correction functions applied to each pixel.

Two major image alignment methods are *Fourier based methods* and *point-based methods*. If the

Figure 2: Output of View Alignment for Panorama Creation

2 input images are related by an affine transformation, then their *Fourier* transforms could be used to find out the transformation parameters [Kuglin and Hines, 1975]. This method is widely used for satellite images and is also robust against noise. However, for registration of non-satellite images, i.e. images taken from hand-held cameras, the space of affine transformations is too small to represent the distortions caused by the depth complexity in real world scenes. This gives rise to *point based* methods. In this class, one needs to specify (or compute) a large number of corresponding points across the 2 views. These corresponding points serve as samples of the final transformation function that needs to be computed. If the transformation model is known beforehand, then the corresponding points could be used to find the transformation parameters. Otherwise, the function is approximated to be a linear/polynomial function interpolating the different points and having values equal to their corresponding disparities. Automatic image registration is difficult in this class since it involves robust feature detection and matching across views.

Image registration using the above methods work well if the disparities involved for each pixel is very small. However, for *widely spaced* cameras, the images are no longer affine transformations of one other causing *Fourier methods* to fail. Moreover, the assumption of linear/polynomial interpolation of disparities also breaks down if there is a lot of depth changes in the scene. Image registration between widely spaced cameras essentially involves solving the stereo problem in computer vision.

This project aims to perform view alignment across *widely spaced* cameras. However, the 3D world being captured (a meeting room) is assumed to be simple, which makes automatic image reg-

istration possible. Moreover, by assuming simple models, we are able to perform real time alignment for 3 sets of cameras using just one PC powered by Intel Xeon 3.4GHz, 3GB RAM running Windows XP.

The paper is divided into 4 sections. Section 2 describes the *Fourier-based* and *point-based* methods of image registration with some examples. Section 3 describes the setting of our meeting room and the assumptions we make for performing image registration. Section 4 describes the actual working of the system and some results. Section 5 concludes the report and suggests directions for future work.

2 Image Registration Methods

2.1 *Fourier Based Methods*

If the transformation between 2 input images can be represented by a similarity transformation (rotation, translation and scale), then these parameters can be computed using the *Fourier transform* of the input images. Operating in the frequency domain increases the robustness of the parameter estimation against noise. Moreover, the computation can also be made fast using FFT which has been widely studied and used.

The main reason for the robust computation of parameters in the frequency domain is the ease of computation of translation. Consider 2 signals $I_1(x,y)$ and $I_2(x,y)$ such that

$$I_1(x,y) = I_2(x-d_x, y-d_y)$$

Then, their corresponding *Fourier transforms* will be related by

$$F_2(\omega_x, \omega_y) = e^{-j(\omega_x d_x + \omega_y d_y)} F_1(\omega_x, \omega_y)$$

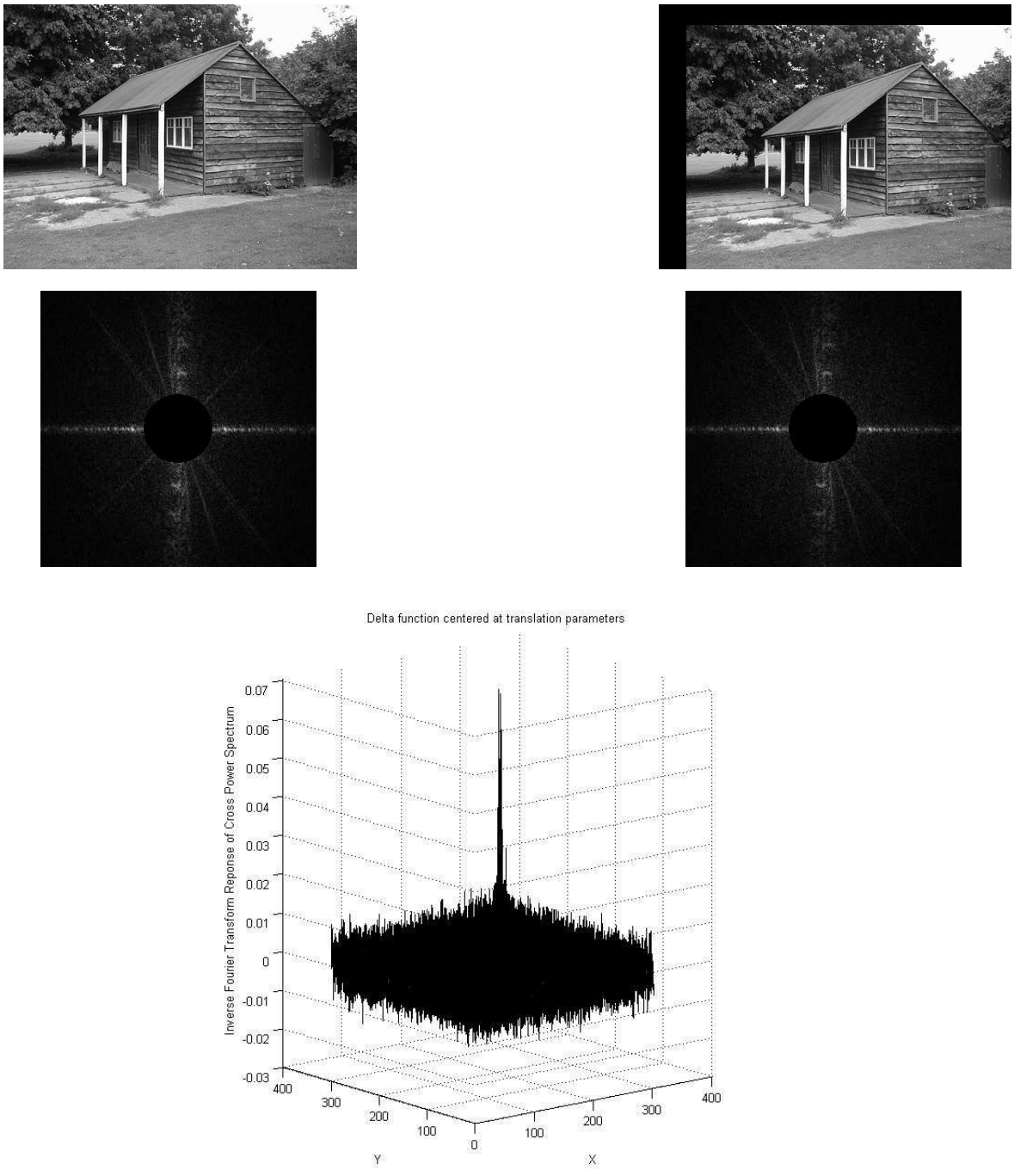

Figure 3: Images showing 2 images related by pure translation, their magnitude spectrum and the *inverse Fourier transform* of the cross-power spectrum term.

Thus, the two signals have the same *Fourier* magnitude but a phase difference which is related to the translation parameters. This phase difference is equivalent to the phase of the cross-power spectrum,

$$\frac{F_1(\omega_x,\omega_y)F_2^*(\omega_x,\omega_y)}{|F_1(\omega_x,\omega_y)F_2^*(\omega_x,\omega_y)|} = e^{(\omega_x d_x+\omega_y d_y)}$$

where $*$ stands for complex conjugate operator. The inverse *Fourier transform* of this phase difference is a delta function centered at the translation point which solves for registration.

2.1.1 Extracting rotation parameters

Consider 2 signals $I_1(x,y)$ and $I_2(x,y)$ which are related by a rotation θ_0 and translation parameters $[d_x,d_y]$. Then,

$$I_2(x,y) = I_1(xcos\theta_0 + ysin\theta_0 - d_x, -xsin\theta_0 + ycos\theta_0 - d_y)$$

Their *Fourier transforms* will be related by

$$F_2(\omega_x,\omega_y) = e^{-j(\omega_x d_x+\omega_y d_y)}F_1(\omega_x cos\theta_0 + \omega_y sin\theta_0, -\omega_x sin\theta_0 + \omega_y cos\theta_0)$$

Thus if we just see the magnitude spectrum of the 2 signals, then we get

$$M_2(\omega_x,\omega_y) = M_1(\omega_x cos\theta_0 + \omega_y sin\theta_0, -\omega_x sin\theta_0 + \omega_y cos\theta_0)$$

which means, that they are related by a simple rotation. Thus, if we transform these images into polar coordinates (r,θ), they will be related by simple translation which can be estimated using the above method.

$$M_2(\rho,\theta) = M_1(\rho,\theta-\theta_0)$$

Once the rotation parameters are determined, the images are corrected to nullify the rotation and

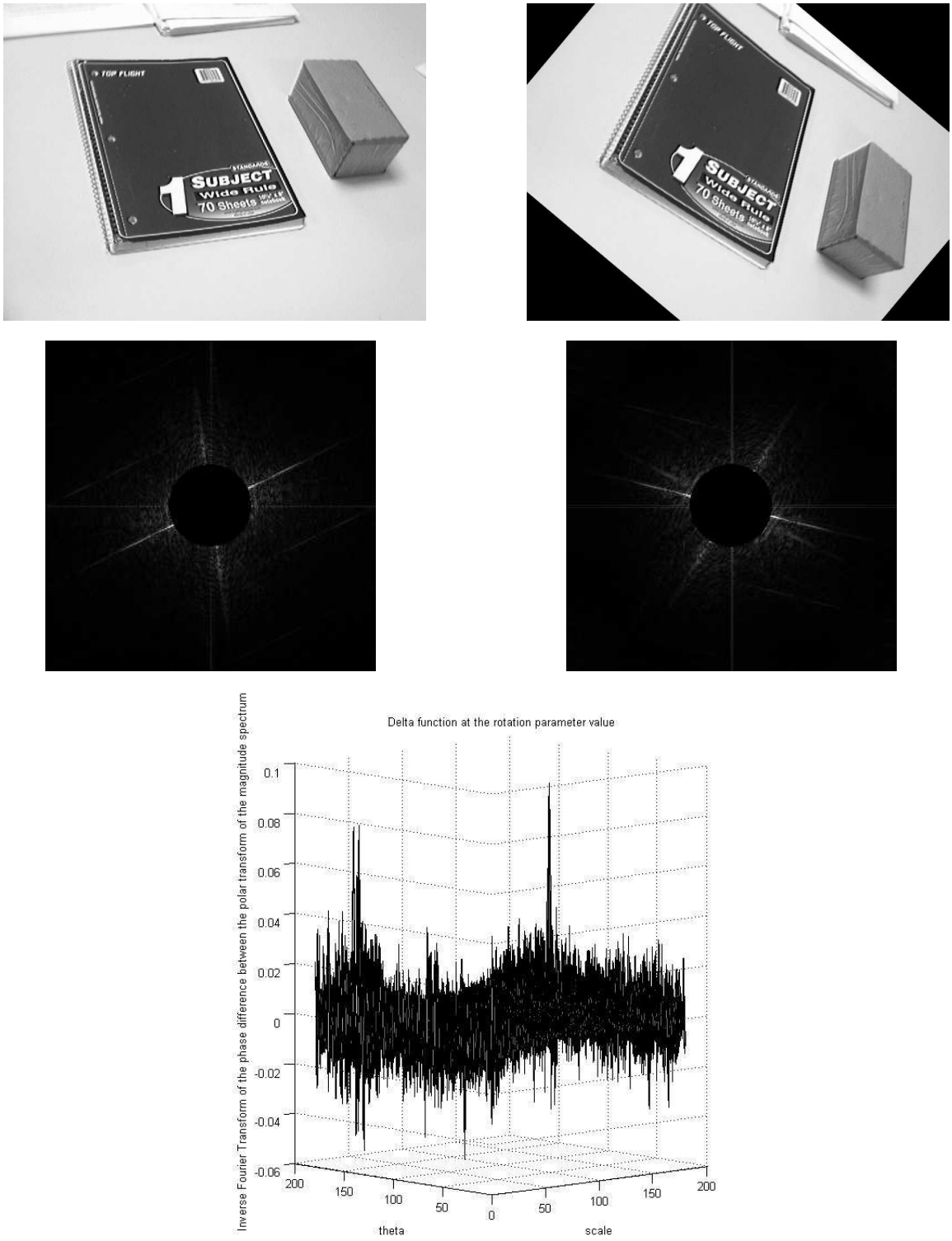

Figure 4: Images showing 2 views related by a pure rotation. Note that their *Fourier magnitude transforms* are related by rotation which is computed using the previous method by first transforming to the *polar* domain.

the translation parameters are computed as in the previous method.

2.1.2 Extracting scale parameters

If the 2 images are related by translation $[d_x; d_y]$, rotation θ_0 and an isotropic scale parameter ρ , then

$$I_2(x,y) = I_1(\rho(xcos\theta_0 + ysin\theta_0) - d_x, \rho(-xsin\theta_0 + ycos\theta_0) - d_y)$$

and thus their corresponding *Fourier transforms* are related by

$$F_2(\omega_x, \omega_y) = e^{-j(\omega_x d_x + \omega_y d_y)} F_1(\frac{(\omega_x cos\theta_0 + \omega_y sin\theta_0)}{\rho}, \frac{(-\omega_x sin\theta_0 + \omega_y cos\theta_0)}{\rho})$$

Thus, their magnitude spectrum in *polar coordinates* are related by

$$M_2(r,\theta) = M_1(\frac{r}{\rho}, \theta - \theta_0)$$

Furthermore, if we convert it into *log-polar* coordinates, we get

$$M_2(log(r),\theta) = M_1(log(r) - log(\rho), \theta - \theta_0)$$

which is again equivalent to computing the translation parameters.

2.2 *Point Based Methods*

Registration can be more generally performed by sampling the registration functions at a finite number of points. Although, the methods discussed in the previous section are very robust and work well, the class of transformations that they cover is very small and simplistic. In general, it is unlikely that the 2 views will be related by a *linear* transform like rotation, translation or shear. Thus, *point based* methods are used for performing registration in this case.

These methods are a part of a bigger subset of methods which are based on feature matching. Matching features across the 2 images gives us information on the final warping function. As discussed in [Brown, 1992], a registration method comprises of the following steps:

- feature space
- similarity metric
- search space and strategy
- transformation function

Feature space refers to the image properties that would be used to perform matching. Some of the examples are points, edges, contours, patches, line intersections etc. Based on the sensor properties, scene conditions and efficiency requirements of the image registration system, different features are used. In this project, we choose *points* to be the features matched.

Similarity metric refers to the closeness function which is used for feature matching. The simplest distance measure for points in images taken from hand-held cameras is *sum of squared differences*. However, due to noise, this measure is not very robust. Robustness is increased by computing the distance between the neighborhoods of the points rather than just the points themselves. We use this in our project for feature matching.

Search space and strategy is concerned more about the computation cost of performing feature matching. The strategy could be to assume certain bounds on the level of distortion across the 2 views. For instance, given a point in the first image, the corresponding point in the second image could only lie on a line in the second image. If we assume that the camera has undergone motion in some particular direction, say v , then the search could be restricted to a band in the second image about the line containing the concerned feature point in the direction v. The size of the band is related to the amount of distortion / camera motion that has occurred between the 2 images. In this project, this strategy has been used and the direction v is assumed to be the horizontal direction.

The transformation function characterizes the distortion between the two images. The actual function in real scenes containing many objects of varying depths, captured by cameras not having a common center of projection is complicated. In fact, the function is discontinuous with object boundaries serving as lines of discontinuity. Thus automated computation of this complicated function is equivalent to solving the vision problem of segmentation and correspondence which is very hard. Thus, approximate transformation functions which roughly map the good feature points to their corresponding positions is usually sufficient for registration purposes. In this project, the transformation function to be computed is chosen to be a projective transformation. The assumption is usually valid if the feature points are very distant compared to the distance between cameras.

The figure shows the application of these steps towards registering 2 views of a lab scene. The *Red* boxes show the detected features while the *Blue* lines show the disparities. Note the rotational nature of these disparities which correspond to the rotation between the 2 images. The features are matched using *neighborhood correlation* and the projective transformation relating the 2 images is computed. The combined image shows seams along the boundaries where registration took place.

3 Meeting Room Application

In this section, we describe the system developed for performing view enhancement of an omnidirectional camera placed in a meeting room. The main component of this system is an image registration module which combines images (information) obtained from other cameras placed in the room. More specifically, the cameras used for view enhancement are placed on the laptops of users involved in the meeting. The system works in real-time making adjustments to the transformation function as the laptop position and orientation changes.

Figure 5: *Lab Scene*: Input images, images with detected features and disparity, final image

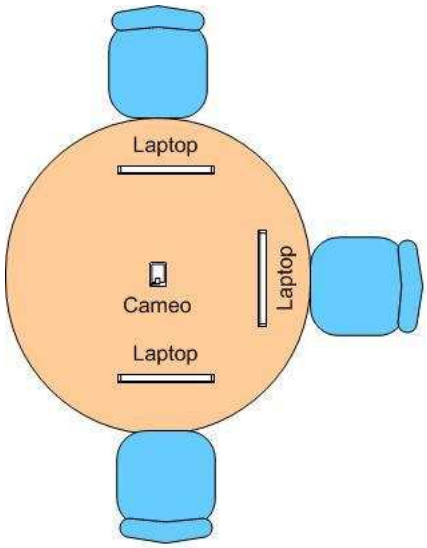

Figure 6: Meeting Room Setting

3.1 Room setting and problem formulation

The room is set up for sensing a meeting among 3 people. It consists of one table and 3 chairs for the people attending the meeting. At the center of the table is placed an omnidirectional camera. This camera, called *Cameo*, is a product from CMU for providing a global sensing of the activities in a meeting room. The camera consists of 4 Firewire cameras connected via a *daisy chain.* Only 3 of these cameras are actually involved in the experiment corresponding to the 3 people sitting in front of them.

Global sensing of activities becomes difficult if the people involved in the meeting are carrying laptops and their views are blocked by the laptop screens. In order to retrieve this blocked information, cameras are fitted on the laptop monitors. These cameras are also Firewire cameras. Thus, the room comprises of an omnidirectional camera placed at the center of the table and 3 laptop cameras providing local views of the people taking part in the meeting.

The main problem in this setup is that information is being retrieved from 4 sensors as separate

video streams (*Cameo* and the 3 laptop cameras). However, the event they represent is the same. Thus, we want to combine these separate sensor information into one. Since the requirement of the additional sensors (laptop cameras) was because the views of *Cameo* were blocked by the laptops, we need to replace the blocked portions of the images by the views obtained from the laptops. Thus, we come down to the problem of *image registration* from widely spaced cameras.

We register the images separately for each view. The *Cameo* actually comprises of 4 cameras, three of which are involved in the problem. The view blocked for each of these cameras is due to exactly one laptop placed in front of it in the scene. Thus for registration, we just need to solve the problem for each pair independently.

Registration of views from widely spaced cameras for a general scene is a hard problem in computer vision. An exact solution to this problem, essentially provides the solution to the *stereo* problem which is still an active area of research. In this project, we propose an approximate solution which makes certain assumptions about the scene.

We use *point-based* techniques of image registration to perform the view alignment. Since these views are not simple affine transformations of one another, *Fourier transform* based techniques cannot be used. As discussed in the previous sections, *point based* methods involve *feature extraction* and their subsequent *matching*. The quality of the registration relies heavily on the quality of these 2 operations. We discuss the potential problems in the application of this method and the assumptions we make to solve those problems in detail, in the next section.

3.2 Algorithmic Design and Assumptions

In this section, we describe the algorithmic details of the system developed for performing the registration process. First, we discuss some of the problems we would encounter, if we were to directly simply apply the *point-based* registration method. Then we discuss the simplifying assumptions we made to solve those potential problems and make the problem more tractable. We then describe the

general strategy used to carry out the registration task.

3.2.1 Potential problems with simple *point-based* registration

Feature matching is the most important component of the registration process. For a good alignment, we need a large number of common features between the *Cameo* view and the laptop view. However, in the current scenario, the area in the images which are viewed by both the cameras is small. Moreover, most of the features in the laptop view would come from the highly textured region occupied by the person sitting in front of it. This adds additional complexity to the feature matching process. The matching takes place across different sensors, which might have a totally different hue/saturation value associated with them. This makes matching between corresponding points in the two images difficult. Also, the laptops change their position and orientations during a meeting and so, the registration should be fast enough to capture these changes and incorporate them in the final registered views. Finally, there is also a concern about the class of transformation function, we choose for performing the registration. The complexity of this function would depend on the number of depth varying objects present in the background that need to match across the 2 views. In other words, we need a *real-time* registration system which can align views of 3 laptops with the three views obtained from the *Cameo* cameras.

3.2.2 Assumptions

The first problem deals with small overlapping area between the view of the laptop and the view of the *Cameo* camera. This problem is critical because it influences the number of matched features between the two views which plays a big role in proper view alignment. The problem is solved by incorporating, *wide-angled* lenses in the two cameras. With this, the area of *unblocked* view in the two cameras increases, while the *blocked* area remains the same. This leads to more features being detected and matched between the 2 views. The increased view area also contributes towards making

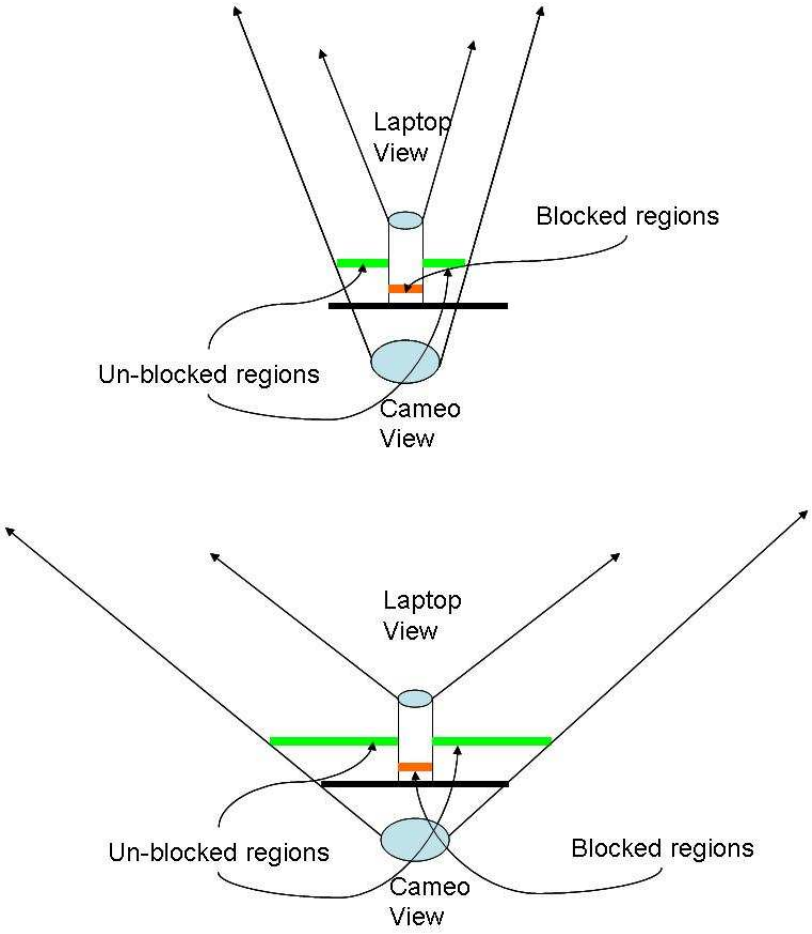

Figure 7: Problem of small overlapping view areas

the effect of extraneous features belonging to the person sitting in front of the laptop, less severe. However, the overall computational cost increases due to increased number of detected features.

The mismatch between the color intensities of the two sensor cameras is also an important problem making the feature matching process very difficult. In order to solve this, a simplifying assumption of constant brightness inside the room was made. It is assumed that intensities across different views of the same camera will be consistent at all times. Since the set up is inside a meeting room, this assumption is fairly reasonable. This property is useful because, instead of matching features across the 2 views directly, we take 2 reference images from the 2 cameras *offline.* The algorithm used subsequently just requires matching of current frame from a camera, with the *reference image* corresponding to that camera. This makes *feature matching* much more robust.

The transformation function between the 2 images is chosen to be a *homography*. In the absence of people in a meeting room, the views from both the *Cameo* camera and the laptop camera comprises primarily of the walls of the room which are planar. Thus, a *homography* is sufficient to describe the transformation between the 2 images. However, in the presence of people this is not true. Since anyways, people are not visible from the *Cameo* camera, a homography would not cause any misalignment between images of people. Background features which belong to the walls of the meeting room would be warped correctly onto the second view. Another reason for choosing *homography* as our transformation function was that computing it is less costly and thus it makes building a real-time system more plausible.

Walls in a meeting room are usually plain and textureless. This is a potential problem because we need to extract a number of features in the background, so that feature matching and hence image registration is possible. Hence, we make the assumption of a textured background.

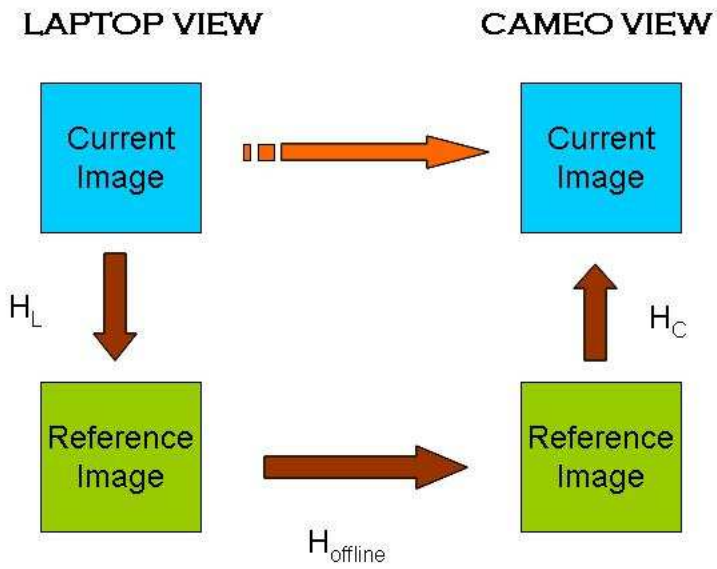

Figure 8: Schematic diagram showing the transformation (homography) pipeline

3.2.3 Algorithmic Design

In this section, we describe the procedure for performing the registration across the 2 views. The key for this procedure is the use of *reference images* obtained offline from the 2 cameras. As described in the previous section, views taken from the same sensor will have similar internal parameters like hue, saturation etc. Thus, corresponding points will have similar image characteristics and this makes feature matching robust.

The homography between the reference views $H_{offline}$ are computed offline. A user is made to mark corresponding points in the 2 views and the homography is computed by a least squares fit using *singular value decomposition.*

The real-time component of the system comprises of computing H_L and H_C for each of the 3 pairs of views. Features from the reference images are computed just once and stored. Now for every frame, new features are computed in both the cameras and matched with respect to the reference features. We use the *cvGoodFeaturesToTrack ()* function of **OpenCV** to perform this computation.

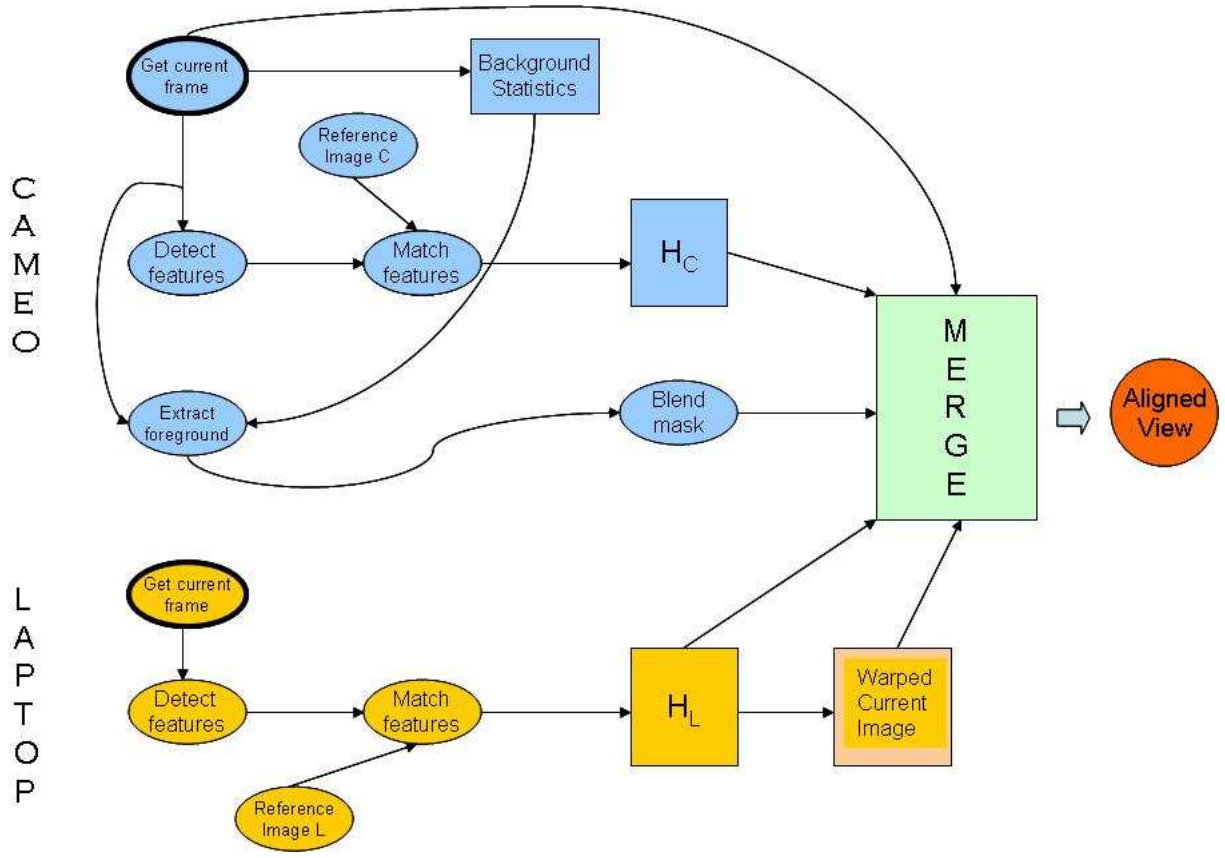

Figure 9: Detailed System Design

Matching is done using a normalized sum of squared differences in a window around the tracked feature. From these matched features, the homographies H_L and H_C for the laptop and cameo cameras are computed respectively. By combining these three homographies, H_L, $H_{Offline}$, and H_C, the required transformation between the current frames of the laptop camera and the cameo camera is computed and thus, view alignment takes place.

Figure 10: Background Subtraction Module: Background image, current frame, foreground blob

4 Detailed System Design

In this section, we describe details of the working of the view alignment system. The description is made separately for the *Cameo camera* and the *laptop camera* subsystems. The output from both these systems are merged to provide with the final image.

4.1 *Cameo camera* subsystem

The *Cameo camera* view contains the blocked (by laptops) regions of the meeting room. Thus, the aim of this system is to provide with the necessary portions in the scene which need to be replaced. Moreover, direct replacement of pixels will lead to artifacts (seams in the image) since the intensity profiles of this camera is different from the *Laptop camera*. This artifact is corrected for using a blending operation between the new view and the relevant portions of the *Cameo* image. Coefficients of this blending operation are also computed in this subsystem.

The first important component of this subsystem is the Background Subtraction module. Necessary portions of the scene which are to be a part of the final view comprises of the unchanged background. The *background statistics* are learnt for a couple of hundred of frames. These statistics comprise of the mean intensity and the standard deviation. We assume the color pixels to vary in accordance with a Gaussian distribution. As the user arrives with the laptop and places it in front of

the camera, the Background Subtraction module clusters together the foreground pixels into one big foreground blob. Pixels which have not been classified as *foreground* are copied to the final image as it is. The output of this module, apart from the unchanged background is also a mask, which is 1 where there is foreground and 0 where there is background.

The second important component of this subsystem is the calculator of H_C. As described in the previous section, this value is essential for computing the final transformation. Since the *Cameo* is usually stable on the center of the table, H_C is only computed every $20th$ frame.

The final component of the system blurs the mask produced by the Background Subtraction module. The mask stores the contribution of the *Cameo camera* image and the *Laptop camera* image in the final output. Pixels which have value 1 have full contribution from the latter camera while pixels with value 0 have full contribution from the former sensor. Note that before blurring, the foreground pixels have values 1 indicating that their values will come from the *Laptop camera* image. The merge of the 2 images without blurring would produce a sharp seam at the boundary of the foreground blob. To make this boundary more diffuse, blurring is performed which makes the sudden transition from 1 to 0 at the boundaries to a more gradual descent thereby producing a smooth blend.

4.2 *Laptop camera* subsystem

The *Laptop camera* view consists of information that will fill in the blocked regions of the *Cameo* view. For every frame, features are detected and matched with a reference image as discussed in the Section 3.2.3 . Using these matched features, the homography between the current frame and the reference frame is computed. The frame is then warped according to this homography and the resultant image is directly fed into the *Merger* (see figure).

Figure 11: View warped with respect to reference view: Reference Image, Current frame, Warped image

Figure 12: Two views merged to give the final output: *Cameo* image, *Laptop* image, Final image

4.3 Merging

The output of the above two subsystems are merged to produce the final output. The *Cameo camera* subsystem provides with 3 inputs: current blocked frame, homography with respect to its reference view and a map containing the blend coefficients. The *Laptop camera* subsystem provides with 2 inputs: warped version of current frame and the homography with respect to its current view. The corresponding pixels (computed using the homographies) in the 2 images are then added using the blend map.

5 Results and Future Work

In this section, we discuss the results obtained from our system and possible areas of future work. These results are displayed in the figure shown. The alignment between the background features is correct because the background is planar. However, the foreground (person) is distorted. The reason is because we are using the same transformation function for the foreground as we used for the background. Ideally, we should use a different warp function taking into account the 3D details of the person. However, since our aim in this project is to perform rough registration so that activities going behind the laptop become visible in *real time*, this level of distortion can be tolerated.

The homography is computed for every frame of the *Laptop camera* image. This computation depends on the matched features. However, the set of features detected for every frame is different due to noise and lighting effects. Moreover, for fast movements of the laptop, the image obtained is blurred making feature detection even more difficult. Finally, the algorithm used for computing the homography is *Ransac* which might produce a different answer every time it is run on the same set of inputs. As a result, the transformation computed is not the same for every frame. This causes instability in the warping. To reduce this instability, we try to restrict a newly computed homography value to a range near the current value of homography.

The system runs at 2 - 3 frames per second. The bulk of the time is spent in computing the matches. Even though the matching computation was not totally exhaustive (a crude epipolar constraint was used), the frame rate is still below what one may desire. In order to increase frame rate, one option is to reduce the number of detected features. The drawback of this will be less reliable image registration. If we choose only the first N features out of the M detected ones, then the probability of the 2 sets of detected features having common world points will be even lower. Moreover, less number of feature points also makes the estimation less robust. Robust feature detectors like SIFT could be used for more reliable detection. Also, detected features are placed relatively fixed with respect to each other. This information could be used for better feature matching. Both of these

Figure 13: Results

could be part of future work in this area.

Our system makes some simplifying assumptions about the choice of background. These include, assumption of a planar surface, and presence of textured regions. Since the transformation is to work in real-time, homography seems to be a good choice for fast computation. Moreover, the procedure is point-based and so assumption of textured background is also important. Future work could involve replacing textured regions by robust markers that could be very easily placed on the walls. Also proposing more complicated transformation models and their fast computation would help in removing the planarity assumption of the background and would incorporate a more appropriate warping for the foreground (person).

References

[Brown, 1992] Brown, L. (1992). A survey of image registration techniques. *ACM Computing Surveys*, 24(4):325–376.

[Guillox, 1986] Guillox, Y. (1986). A Matching Algorithm For Horizontal Motion, Application To Tracking. In *8th International Conference on Pattern Recognition*, Paris, France.

[Kuglin and Hines, 1975] Kuglin, C. and Hines, D. (1975). The Phase Correlation Image Alignment Method. In *Intl. Conf. Cybernetics and Society*, pages 163–165.

[Merickel, 1988] Merickel, M. (1988). 3d Reconstruction: The Registration Problem. *Computer Vision, Graphics and Image Processing*, 42(2):206–219.

[Pelizzari et al., 1989] Pelizzari, C., Chen, G., Spelbring, D., Weichselbaum, R., and Chen, C. (1989). Accurate Three-Dimensional Registration of CT, PET and/or MR Images of the Brain. *Journal of Computer Assisted Tomography*, 13:20–26.

[Ratib et al., 1988] Ratib, O., Bidaut, L., Schelbert, H., and Phelps, M. (1988). A New Technique For Elastic Registration of Tomographic Images. *IEEE Proceedings SPIE: Medical Imaging II*, 914:452–455.

[Uyttendaele et al., 2003] Uyttendaele, M., Criminisi, A., Kang, S., Winder, S., Hartley, R., and Szeliski, R. (2003). High-quality image-based interactive exploration of real-world environments. *Computer Graphics and Applications*, 24(3).